uc3m | Universidad Carlos III de Madrid

Grado Universitario en Sistemas de Tecnologías de Telecomunicación

2017-2018

Trabajo Fin de Grado

"Análisis e implementación de un sistema de recomendación para la lista de la compra"

Marta Criado González

Tutor/es

Vanessa Gómez Verdejo

Leganés, 2018

RESUMEN

En la presente memoria se lleva a cabo el estudio y la implementación de un sistema de recomendación capaz de aconsejar a los usuarios sobre los productos que mejor se ajustan a sus necesidades a la hora de hacer la compra.

Para comenzar, se analizarán los distintos tipos de sistemas de recomendación (SR) y las técnicas que se han utilizado desde que estos sistemas salieron a la luz. Debido al auge de esta tecnología, tanto las técnicas como los algoritmos y la manera de conseguir las bases de datos han sufrido un cambio muy abrupto en unos pocos años. Este desarrollo exponencial se verá plasmado en el proyecto puesto que veremos cómo los algoritmos matemáticos han ido evolucionando hasta ser capaces de pensar como el propio usuario.

Previamente a la implementación final de este sistema se hará un análisis exhaustivo de la base de datos extraída de Instakart para poder saber así si es válida y como poder sacarle la máxima funcionalidad posible.

Una vez este análisis está completo, se procede a la exposición del caso práctico. Mediante este sistema de recomendación los usuarios podrán saber qué artículos son los que se ajustan más a sus gustos y necesidades a la hora de hacer la compra.

Se hará uso de un filtrado colaborativo (FC) con información implícita mediante técnicas de ***machine learning.***

Palabras clave: sistema de recomendación, filtrado colaborativo, machine learning.

ABSTRACT

This project exposes the implementation of a recommender system with collaborative filtering (CF) techniques capable of giving recommendations to users about the products that fit better to their necessities when doing the shopping list.

Firstly, the four types of RS (Recommender Systems) will be analyzed deeply and the different techniques that have been used since they were born. Due to the huge impact of this technology, the techniques used ,the algorithms and the way of picking the data from the users have suffered an abrupt change in a short period of time. This exponential development will be captured in the project as it will be seen that the mathematical algorithms have been evolving in such a way that nowadays it recreates human's way of thinking.

Before the implementation of the RS, a database analysis will be done. This chapter is crucial because this is where we can model and know more about our data. Finally, the practical case of the development of a RS will be explained with all type of details. This RS will be developed with machine learning techniques and implicit data.

Keywords: machine learning, recommender systems, collaborative filtering

DEDICATORIA

Deseo expresar mi agradecimiento a todos los que me han acompañado y ayudado durante la elaboración del trabajo, en particular a mi tutora, por haberme orientado y facilitado todas las fuentes de conocimiento necesarias, y a mi familia por apoyarme.

“El hombre que pretende verlo todo con claridad antes de decidir nunca decide”,
H-F. Amiel

INDICE DE CONTENIDO

RESUMEN **3**

ABSTRACT **4**

1 INTRODUCCION **15**

1.1 Motivación del proyecto 16

1.2 Marco socioeconómico 17

1.3 Objetivo concreto de este estudio 20

1.4 Contenido de la memoria 21

2 ESTADO DEL ARTE: SISTEMAS DE RECOMENDACIÓN **23**

2.1 Introducción 23

2.2 Clasificación de los sistemas de recomendación 23

2.2.1 Filtrado colaborativo 24

2.2.2 Sistemas de recomendación basados en contenido 26

2.2.3 Sistemas de recomendación basados en conocimiento 30

2.2.4 Sistemas de recomendación híbridos 32

3 ANALISIS DE LA BASE DE DATOS **34**

3.1 Estructura inicial 34

3.2 Análisis de puntuaciones para el sistema de filtrado colaborativo 36

3.2.1 Análisis de los pedidos por parte de los usuarios 37

3.3 Base de datos para el sistema de recomendación basado en contenido 42

3.4 Separación de los conjuntos de entrenamiento, test y validación 43

3.5 Obtención de la matriz de puntuaciones 45

3.6 Solución al problema de la dispersión de datos 47

3.7 Análisis del conjunto de entrenamiento 48

4 ELEMENTOS DE UN SISTEMA DE RECOMENDACIÓN CON FILTRADO COLABORATIVO **51**

4.1 Tipos de puntuaciones para filtrado colaborativo [9] 51

4.2 Algoritmos para el sistema de recomendación con filtrado colaborativo 52
4.2.1 Métodos de vecinos cercanos [24] 52
4.2.2 Modelos de factores latentes 53
4.2.3 Algoritmos de aprendizaje 57
4.3 Medidas para la precisión de las predicciones 59
4.3.1 Métricas para la ayuda en la toma de decisiones 59
4.3.2 Métricas para la predicción de la precisión en puntuaciones 60
5 IMPLEMENTACIÓN DEL SISTEMA DE RECOMENDACIÓN 65
5.1 Implementación del algoritmo ALS manualmente 65
5.1.1 Ajuste de parámetros 66
5.1.2 Resultados del sistema de recomendación 70
5.1.3 Resultados del sistema real para los dos usuarios con mayores puntuaciones 72
5.1.4 Recomendaciones a usuarios y productos similares 73
6 GESTIÓN DEL PROYECTO 75
6.1 Planificación del proyecto 75
6.2 Análisis de costes 75
6.2.1 Personal 75
6.2.2 Presupuesto total 77
7 CONCLUSIONES 78
7.1 Conclusión 78
7.2 Líneas futuras de trabajo 79
BIBLIOGRAFÍA 80
ANEXO I. ACRÓNIMOS 85
8 EXTENDED SUMMARY 86

INDICE DE FIGURAS

Figura 1.1. Imagen de un producto y sus recomendaciones extraída del servicio de Amazon Fresh 18

Figura 1.2. Gráfica con la evolución del número de compras en supermercados mediante sus tiendas online [7] 19

Figura 1.3. Imagen de ejemplo de la compra de un producto en la página web www.lafruteria.es 19

Figura 1.4. Sección de puntuaciones en la página web de www.lamejornaranja.com .. 20

Figura 1.5. Sección de productos similares recomendados en la página web de www.lamejornaranja.com 20

Figura 2.1. Esquema del funcionamiento del filtrado colaborativo [10] 24

Figura 2.2. Esquema del funcionamiento de los SR basados en contenido [10] 27

Figura 2.3. Espacio de vectores del ejemplo para SR basados en contenido 30

Figura 2.4. Diagrama de flujo recomendación case-based [16] 31

Figura 2.5. Árbol de decisión [17] 32

Figura 2.6. Diagrama de flujo sistema de recomendación híbrido [18] 32

Figura 3.1. Estructura primer fichero 35

Figura 3.2. Estructura segundo fichero 35

Figura 3.3.Conjunto de datos 36

Figura 3.4. Cantidad de pedidos por usuario 37

Figura 3.5. Pedidos por días de la semana 38

Figura 3.6. Mapa de calor frecuencia de pedidos en el día de la semana frente a horas del día 38

Figura 3.7. Frecuencia con la que un usuario vuelve a hacer un pedido en el conjunto de entrenamiento 39

Figura 3.8. Frecuencia con la que un usuario vuelve a hacer un pedido en el conjunto de prior 39

Figura 3.9. Porcentaje de productos que tienen valor nulo en el conjunto prior 41

Figura 3.10. Repetición de ordenes con respecto a cada departamento 41

Figura 3.11. Recuento del total de productos por pasillo. En el eje de abscisas se encuentra el recuento de productos en cada pasillo, en el de ordenadas el nombre de cada pasillo 42

Figura 3.12. Distribución de productos por departamentos 43

Figura 3.13. Ejemplo para la explicación de la validación cruzada con 4 iteraciones .. 45

Figura 3.14. Diagrama de caja de la columna de puntuaciones inicial 46

Figura 3.15. Histograma del número de apariciones de cada usuario en el conjunto inicial de datos 47

Figura 3.16. Histograma del número de apariciones de cada producto en el conjunto inicial de datos 47

Figura 3.17. Histograma de la distribución del número de apariciones de los usuarios en el conjunto de datos filtrado 48

Figura 3.18. Histograma de la distribución del número de apariciones de los productos en el conjunto de datos filtrado 48

Figura 3.19. Histogramas de la distribución del número de usuarios y productos en el conjunto de entrenamiento 49

Figura 3.20. Diagrama de caja con la distribución de las puntuaciones en el conjunto de entrenamiento 50

Figura 3.21. Diagrama de caja con la distribución de la media de puntuaciones dada por cada usuario en el conjunto de entrenamiento 50

Figura 4.1. Ejemplo factores latentes Netflix Price [1] 54

Figura 4.2. Descomposición en valores singulares [25] 55

Figura 4.3. Ejemplo factorización matricial [25] 55

Figura 4.4. Diagrama de factores latentes para el caso de estudio 56

Figura 4.5. Muestra gráfica de los conjuntos que se utilizan para calcular la precisión y el recall 59

Figura 4.6. Ejemplo para la explicación de las métricas de precisión y recall 60

Figura 4.7. Ejemplo para la explicación de la métrica de precisión media en dos recomendaciones distintas [33] 62

Figura 4.8. Ejemplo para explicar la métrica DCG 63

Figura 4.9. Diferencia de pendiente entre el mAP y nDCG [35] 64

Figura 5.1. Evolución del error cuadrático medio con el número de iteraciones 65

Figura 5.2. Evolución de la métrica mAP vs. parámetro de regularización (25 iteraciones) 67

Figura 5.3. Evolución de la métrica nDCG vs. parámetro de regularización (25 iteraciones) 67

Figura 5.4. Evolución de las métricas nDCG y mAP con respecto a los distintos valores de factores latentes para 20 iteraciones del algoritmo 68

Figura 5.5. Evolución de las métricas nDCG y mAP con respecto a los distintos valores de factores latentes para 25 iteraciones del algoritmo 68

Figura 5.6. Evolución de las métricas nDCG y mAP con respecto a los distintos valores de factores latentes para 30 iteraciones del algoritmo 68

Figura 5.7. Representación de las recomendaciones que se obtienen del ALS 73

Fig. 8.1. Functionality diagram of the RS algorithm 91

INDICE DE TABLAS

Tabla 2.1.Resultados de la técnica TF-IDF para el ejemplo propuesto 29

Tabla 2.2. Métodos de hibridación [19] 33

Tabla 3.1. Número de elementos que pertenece a cada conjunto 40

Tabla 5.1. RESULTADOS DE LAS MÉTRICAS MAP Y NDCG EN UNA LISTA DE RECOMENDACIÓN TOP-3 SOBRE EL CONJUNTO DE VALIDACION..... 69

Tabla 5.2.RESULTADOS DE LAS MÉTRICAS MAP Y NDCG EN UNA LISTA DE RECOMENDACIÓN TOP-4 SOBRE EL CONJUNTO DE VALIDACION..... 70

Tabla 5.3. RESULTADOS DE LAS MÉTRICAS MAP Y NDCG EN UNA LISTA DE RECOMENDACIÓN TOP-5 SOBRE EL CONJUNTO DE VALIDACION..... 70

Tabla 5.4. RESULTADOS DE LAS MÉTRICAS MAP Y NDCG EN UNA LISTA DE RECOMENDACIÓN TOP-10 SOBRE EL CONJUNTO DE VALIDACION..... 70

Tabla 5.5. RESULTADOS DEL SISTEMA DE RECOMENDACIÓN EN EL CONJUNTO DE TEST CON LOS 3 PRIMEROS ELEMENTOS DE UNA LISTA DE RECOMENDACIÓN..... 71

Tabla 5.6.RESULTADOS DEL SISTEMA DE RECOMENDACIÓN EN EL CONJUNTO DE TEST CON LOS 4 PRIMEROS ELEMENTOS DE UNA LISTA DE RECOMENDACIÓN..... 71

Tabla 5.7.RESULTADOS DEL SISTEMA DE RECOMENDACIÓN EN EL CONJUNTO DE TEST CON LOS 5 PRIMEROS ELEMENTOS DE UNA LISTA DE RECOMENDACIÓN..... 71

Tabla 5.8. RESULTADOS DEL SISTEMA DE RECOMENDACIÓN EN EL CONJUNTO DE TEST CON LOS 10 PRIMEROS ELEMENTOS DE UNA LISTA DE RECOMENDACIÓN..... 71

Tabla 5.9. Resultados del rendimiento de una lista de recomendación para los usuarios más activos 72

Tabla 6.1.Tareas del personal 76

Tabla 6.2.Coste del personal..... 77

Tabla 6.3.Coste total del proyecto 77

1 INTRODUCCION

Los SR surgieron por un simple hecho, los usuarios siempre buscamos una recomendación para tomar una decisión sobre cualquier aspecto. Algunos ejemplos pueden ser la elección de una película por medio de las críticas y los comentarios hacia ella por parte de otros usuarios, cuando un empleado de una tienda nos recomienda un producto…

Para poder copiar este comportamiento que todos tenemos día a día los primeros SR utilizaban unos algoritmos que se basaban en una serie de puntuaciones ya dadas por un conjunto de usuarios anteriormente. Las recomendaciones se hacían en base a usuarios similares al usuario activo o productos similares y la técnica usada se denominan **filtrado colaborativo**. Un claro ejemplo es un SR de películas en el que se sugieren películas que han sido calificadas con puntuaciones altas por parte de usuarios que comparten gustos con el usuario activo.

En el campo del comercio electrónico (CE), los clientes tienen que enfrentarse a un nuevo entorno en el que existen numerosos artículos, en el que las comparaciones entre el precio, la calidad o la marca juegan un papel muy importante en la decisión que van a tomar. Todos estos motivos hacen imprescindible tanto para las empresas como para los clientes un sistema de recomendación que sea capaz de pensar casi como el propio consumidor y recomendarle el producto concreto que está buscando de acuerdo con sus gustos y necesidades. Algunas empresas como Amazon o Netflix ya poseen estos SR en sus páginas web y, como se explicará en el marco socioeconómico, el impacto de estos sistemas supone para las empresas una gran cantidad de ingresos y popularidad.

En este proyecto se hará un estudio y un análisis de los distintos tipos de SR que existen y los algoritmos que se utilizan en cada uno de ellos. Más adelante, y teniendo en cuenta el análisis hecho en los Capítulos 2 y 4, se implementará un sistema de recomendación con filtrado colaborativo que ayude a los usuarios en el momento de hacer la compra.

Esta implementación tendrá un previo análisis de la propia base de datos utilizada para el proyecto, el cual ha sido fundamental para obtener unos resultados óptimos.

1.1 Motivación del proyecto

Dada la gran cantidad de información que tiene a su disposición cualquier usuario en la actualidad, el simple hecho de hacer la compra diaria se convierte en una gran toma de decisiones. Estamos condicionados por los precios de los productos, la feroz competencia entre marcas por hacer que su artículo sea el que más guste al consumidor etcétera…

El propio usuario actúa como un sistema de recomendación al tomar la decisión de qué artículo comprar, con la simple diferencia de que no es un algoritmo matemático el que le está diciendo que elección tomar. Para ayudar a que este proceso sea mucho más cómodo y más fácil para cualquier consumidor, incorporamos los SR.

En los últimos años el interés por los SR ha aumentado de manera significativa. Estos sistemas juegan un papel muy importante en plataformas online de algunas empresas como Amazon, Youtube, TripAdvisor o Ebay. Tal es la importancia de las recomendaciones que Netflix creó un concurso, ***Netflix Price*** *[1]*, para poder mejorar sus recomendaciones a los usuarios y el ganador de esta prueba era agraciado con un premio de un millón de dólares.

Debido al éxito de la implantación de SR en estos portales, el mundo de la alimentación también se ha sumado a esta moda. Muchas de las aplicaciones de las grandes cadenas de supermercados tienen un SR que sugiere artículos que adquirir a los usuarios.

Cualquier empresa busca mejorar sus ventas con la introducción de un sistema de recomendación en su portal web y esta quizás es una de las razones más importantes por las que las empresas implantan estos sistemas. Otra de las ventajas es que se pueden vender productos muy diversos puesto que a cada usuario el algoritmo le recomendará algo que se ajuste a sus preferencias, de esta manera la empresa no está tratando de vender siempre el mismo producto, sino que puede que ofrezca algo muy impopular por el resto de los usuarios pero que sea justo lo que busca un determinado usuario. Esta ventaja está claramente ligada a la satisfacción del cliente y a su fidelidad a la empresa. Si la mayoría

de los usuarios quedan satisfechos con la recomendación que el sistema le ha ofrecido, volverán a entrar en el portal web que estén utilizando y esto hará que se fidelicen más clientes. A su vez esta fidelidad tiene un punto positivo para la empresa puesto que cuanto más se utilice el sistema de recomendación más información acerca de cada usuario le estaremos dando al algoritmo encargado de la decisión, y por tanto cada vez tendrá mejores resultados.

En este proyecto se pretende desarrollar un sistema de recomendación que sea útil para cualquier persona de a pie a la hora de hacer la compra. También ayudará a las empresas que se dediquen a la venta de alimentos a tener más información sobre los clientes. Recomendando productos se intentará conseguir el mayor grado de satisfacción del cliente para poder fidelizarlo y así poder tener más beneficios.

1.2 Marco socioeconómico

El comercio ha sufrido una gran evolución en estos últimos años con la entrada del comercio electrónico. Cada vez son más las empresas que se suman a la tendencia de abrir una tienda en la web en la que los consumidores puedan adquirir directamente un producto y recibirlo en su domicilio sin tener que moverse de la cama.
El auge del comercio electrónico se debe en gran medida a la comodidad que supone el proceso de compra. Mientras que hace unos años ir a comprar a cualquier tienda podía suponer esperar largas colas y perder una gran cantidad de tiempo, en la actualidad eso ya no es necesario. Las nuevas generaciones quieren ahorrar tiempo y hacer todo de la manera más rápida posible sin tener que esperar.
Una de las estrategias que se usa en el CE son los SR. Mediante ellos podemos se puede tanto fidelizar clientes como captar nuevos clientes, lo que supondrá siempre un aumento en las ventas de la empresa.

Según un informe [2] de unos trabajadores de la compañía McKinsey *"el 35 % de los ingresos de Amazon se deben a los SR y casi el 75 % de lo visionado en Netflix proviene también de las recomendaciones"*. Muchas veces la gente comparte su cuenta de Amazon o navega en las mismas páginas webs que otra persona desde el mismo ordenador. Al realizar cualquier de estas dos acciones el SR está captando el comportamiento de dos o

más personas como si fuera el de una sola y, por tanto, está elaborando un perfil erróneo del usuario.

Para evitar este inconveniente unos investigadores del MIT (Massachusetts Institute of Technology) [3,4] están elaborando un sistema que permita diferenciar a los distintos usuarios. Este avance sería un paso más hacía un SR cada vez más inteligente y con el objetivo claro de que este tipo de sistemas cada vez se adapten al pensamiento no lineal que tenemos los humanos.

Debido al éxito de estos SR en plataformas de tiendas online, determinadas empresas dedicadas a la alimentación han decidido implantar estos sistemas en sus portales. Un claro ejemplo de esto es el nuevo servicio que Amazon lanzó en 2016 en Reino Unido y en 2009 en Estados Unidos, **Amazon Fresh** [5]. Esta plataforma sirve para comprar productos frescos y utiliza el mismo SR que Amazon usa en el resto de sus servicios.

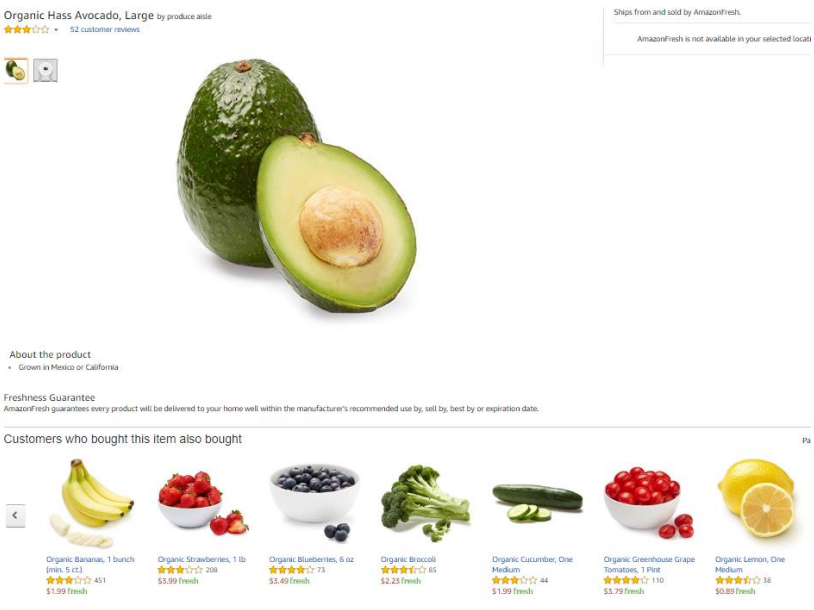

Figura 1.1. *Imagen de un producto y sus recomendaciones extraída del servicio de Amazon Fresh*

En España [6] ha habido una subida en los últimos años en el comercio electrónico, y esto ha sido debido, en gran parte, a los supermercados. En la Figura 1.2 se puede observar como a partir del 2015 el número de transacciones de CE ha sufrido un incremento. Este incremento de ventas está estrictamente relacionado con el desarrollo de las tiendas online de los supermercados, con la implementación de SR como uno de

esos avances. El descenso de las ventas en determinadas épocas del año es debido a que en verano se registra un descenso de compras.

Figura 1.2. *Gráfica con la evolución del número de compras en supermercados mediante sus tiendas online [7]*

Una de las plataformas que más éxito tiene en España es "La Frutería" con una tienda online en la que cada vez que se selecciona un producto para proceder a comprarlo, en la parte inferior de la pantalla aparecen una serie de recomendaciones basadas en productos similares (Figura 1.3). El fundador de esta empresa [8] reconoce "*que conocer más al usuario, saber qué compra y por qué no compra ciertos artículos ha hecho que los beneficios en su empresa se hayan incrementado*".

Figura 1.3. *Imagen de ejemplo de la compra de un producto en la página web www.lafruteria.es*

Otra de las plataformas es "La mejor naranja", en la que existe un apartado de la página de cada artículo para puntuarlo. Estas puntuaciones se utilizarán como información de cada artículo. También contiene un apartado de recomendación de artículos similares basado en el producto que el usuario va a adquirir.

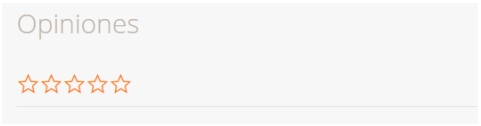

Figura 1.4. *Sección de puntuaciones en la página web de www.lamejornaranja.com*

Figura 1.5. *Sección de productos similares recomendados en la página web de www.lamejornaranja.com*

Una vez explicado el entorno socioeconómico podemos asegurar que, aunque en España haya empresas innovadoras en implantar SR aún falta mucho camino para poder igualar a países como Reino Unido o Estados Unidos. Para que haya un avance es necesario implementar un SR que tenga en cuenta las preferencias de los usuarios y no solo las similitudes entre productos. Cuantas mejores prestaciones ofrezca este sistema, más fidelización obtendremos por parte del cliente y más beneficios tendrá la empresa.

1.3 Objetivo concreto de este estudio

El principal objetivo de este proyecto es el estudio de los SR y más en profundidad de los algoritmos que se pueden utilizar tanto para el filtrado colaborativo como para los sistemas basados en contenido. Los resultados de este estudio se pondrán en práctica en el Capítulo 5 con la implementación de un sistema de recomendación para la lista de la compra como caso práctico y real. A continuación, se enumeran los objetivos concretos:

- Estudio de los distintos tipos de SR, con las ventajas y desventajas de implementación que han tenido en un marco temporal.
- Estudio de los distintos algoritmos que se pueden utilizar para cada tipo de sistema de recomendación. En el proyecto también se analizarán los problemas que pueden surgir al utilizar cada uno de ellos y se expondrá soluciones en caso de que existan.
- Caso práctico con un sistema de recomendación para la lista de la compra en el que se pondrá en práctica toda la teoría anteriormente expuesta con la finalidad de obtener los mejores resultados posibles.
- Implementación propia del algoritmo Alternating Least Squares (ALS).

Dentro del caso práctico del proyecto los objetivos son:

- Obtención de una fiabilidad alta por parte de los resultados
- Correcta adaptación de los datos dados a los algoritmos a utilizar
- Análisis completo de la base de datos que se utiliza

1.4 Contenido de la memoria

El esquema del proyecto se detalla a continuación:

- *Capítulo 1*. ***Introducción.*** Se da una explicación del porqué de la realización del proyecto. Se explica desde el plano socioeconómico la importancia de los SR actualmente y las razones de su aparición. Por último, se exponen los objetivos del proyecto.
- *Capítulo 2*. ***Estado del arte: Sistemas de recomendación.*** Se presentan los diferentes tipos de SR.
- *Capítulo 3*. ***Análisis de base de datos.*** Este apartado consta del análisis de los datos que posteriormente incorporaremos al caso práctico.
- *Capítulo 4*. ***Elementos de un sistema de recomendación con filtrado colaborativo.*** En este capítulo se explican los distintos algoritmos que se pueden usar en un SR con filtrado colaborativo. También se explican los distintos tipos de puntuaciones debido a la importancia de ellas en estos sistemas. Finalmente, se presentarán las métricas para analizar el error de predicción del SR.

- *Capítulo 5.* ***Implementación del sistema de recomendación.*** En esta sección se explica la implementación de la recomendación para la lista de la compra. También se presentan los resultados y los problemas que han surgido a lo largo del proceso.
- *Capítulo 6.* ***Gestión del proyecto.*** Planificación del proyecto con clara diferenciación de tareas entre trabajadores y posteriormente se enumeran los costes del proyecto.
- *Capítulo 7.* ***Conclusiones.*** Se concluye el proyecto y se exponen las mejoras que se podrían desarrollar como futuras líneas de proyecto.

2 ESTADO DEL ARTE: SISTEMAS DE RECOMENDACIÓN

En este capítulo se expondrán los cuatro tipos de SR que existen en la actualidad. Dentro de cada categoría se explicarán las distintas técnicas usadas, así como los problemas que pueden surgir. Existen tres tipos de SR los cuales funcionan con sus propias técnicas y un cuarto tipo, los híbridos, que combinan algunos de los sistemas anteriores para poder formar un sistema "global" en el que cada tipo de sistema arregle los fallos que ocasiona cualquiera de los otros.

2.1 Introducción

En nuestra vida cotidiana se nos presentan continuamente ocasiones en las que tenemos que tomar decisiones. Estas pueden ser la elección de un restaurante, la de una película, un libro etc.… Pero hay una razón por la que estas decisiones se han vuelto cada vez más complicadas, y es la gran cantidad de información que tenemos en nuestras manos.

Los SR surgen de la necesidad de paliar estas dificultades y hacer más fácil la elección del usuario. Mediante una serie de algoritmos podemos conocer los gustos del usuario y recomendarle el producto que se ajuste lo más posible a sus preferencias. Con los avances en tecnología y el paso de los años estos sistemas se han vuelto cada vez más sofisticados y puede parecer que incluso nos conocen.
Multitud de plataformas en Internet y empresas utilizan estas recomendaciones para saber más sobre sus usuarios y para poder ofrecerles un mejor producto y servicio.
A continuación, se expondrán los distintos tipos de SR que existen en la actualidad.

2.2 Clasificación de los sistemas de recomendación

Dentro de los SR existen distintos tipos y técnicas a utilizar. Los dos principales sistemas son el filtrado colaborativo (FC) y el basado en contenido (BC), pero también existen los sistemas basados en conocimiento y los híbridos. Todos estos tipos serán descritos en este capítulo. Para explicar cada uno de los sistemas se va a utilizar un ejemplo de un sistema de recomendación de películas.

2.2.1 Filtrado colaborativo

La idea principal del FC [9] es que si dos usuarios son similares tendrán preferencias igualmente similares. Un punto de especial importancia en un algoritmo de FC son las puntuaciones de los usuarios hacia los distintos productos. Estas calificaciones pueden suponer un problema en la implementación de cualquier SR, en la mayoría de las ocasiones, por su escasez.

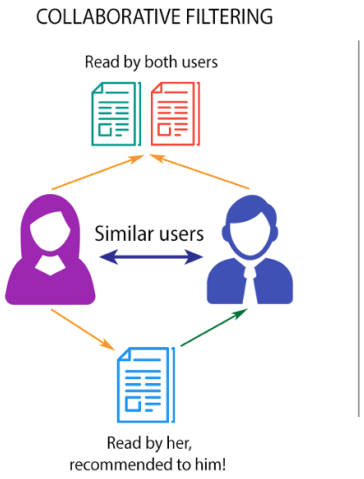

Figura 2.1. *Esquema del funcionamiento del filtrado colaborativo [10]*

El funcionamiento es el siguiente: cuando un usuario A y un usuario B tengan un historial de recomendaciones muy parecido, si el usuario A adquiere un producto nuevo podemos suponer que al usuario B también le parecerá interesante y es altamente probable que lo compre. Por tanto, se recomendará al usuario B comprar ese artículo.

Estos tipos de SR son uno de los más utilizados. En cualquier portal web que contenga alguna recomendación pedirán al usuario una calificación sobre el servicio ofrecido o sobre el producto comprado. Este acto sirve al SR para actualizar la información que tienen de cada usuario y para que el algoritmo pueda aprender. Cuanta más información se tenga de las puntuaciones dadas por cada usuario más fiel será la próxima recomendación y los resultados tendrán un mayor impacto positivo.

Dentro del FC existen dos tipos de aproximaciones: basadas en el usuario (BU) (*user based)* y basadas en los productos (BP) (*item based)*.

2.2.1.1 Basados en usuarios (UB)

La primera aproximación de este tipo de sistemas está basada en los usuarios más cercanos [9,11] (*user-based nearest neighbor*). La idea principal de UB es que, si dos usuarios han puntuado artículos de manera similar en el pasado, tendrán los mismos gustos en el presente.

Esta técnica se explicará mediante el siguiente ejemplo. Supongamos que un usuario *(Marta)* ha visionado las mismas películas que nosotros y las ha puntuado de la misma manera que nosotros, pero hay una película, *Terminator*, que no ha visto. Según el método UB sería lógico decir que, si a nosotros nos ha gustado *Terminator*, a Marta también le gustará. Para aplicar esta lógica se utiliza el algoritmo de Vecinos Cercanos *(Nearest Neighbor*) [11]. Este algoritmo consta de dos partes:

1. Encontrar los K vecinos más cercanos a Marta usando funciones de similitud en las que se mida la distancia entre usuarios. Algunas de las métricas usadas son: la correlación de Pearson, la similitud por coseno *(cosine similarity)* o similitud por coseno ajustada *(adjusted cosine similarity).* Estas métricas se explicarán en más detalle en el capítulo 4.
2. Predecir la puntuación que Marta daría a todos los artículos que los K vecinos calculados han visto pero Marta no. El resultado será el que mejor puntuación predicha tenga. Una de las medidas de predicción que se puede usar es la de la ecuación 2.2 donde *sim* es la función de similitud que se ha usado en el punto anterior, *r* es la puntuación de un usuario y $\bar{r}$ es la media de las puntuaciones de ese mismo usuarios.

Las conclusiones de este método son que si los usuarios han tenido gustos similares en el pasado también los tendrán en el futuro y por tanto las preferencias permanecen estables con respecto al tiempo. Esta técnica se denominada **basada en memoria.**

Algunos de los problemas que surgen con los SR basados en usuarios son el *"cold-start problem"* porque el número de usuarios con puntuaciones es muy bajo, la escalabilidad porque cuantos más usuarios haya más complicado va a ser encontrar los K vecinos y la dispersión de los datos.

2.2.1.2 Basados en productos (IB)

Volviendo al ejemplo del SR de películas, si en vez de fijarnos en las similitudes entre usuarios tenemos en cuenta las similitudes entre productos estaremos utilizando una técnica llamada basado en productos *(item-based)* [9, 11].

Este método tiene cierto parecido a los sistemas basados en contenido, con la diferencia de que la similitud entre artículos se obtiene de las puntuaciones dadas por los usuarios. A diferencia de esto, en los sistemas basados en contenido, que veremos más adelante, la similitud entre artículos se obtiene con las características de cada producto. Las principales ventajas con respecto a los sistemas basados en usuarios son que previene el problema del arranque en frío y que mejora la escalabilidad porque la similitud entre objetos es más estable que entre usuarios.

Recuperando el ejemplo del apartado anterior, si Marta ha dado una calificación alta a "*El Orfanato",* película del género de terror, esta técnica sostiene que se puede suponer que Marta calificará con una puntuación alta a la película "*Rec"* ya que pertenece al género de terror. Esta técnica tiene dos etapas [11]:

1. Calcular la similitud entre los artículos "*Rec" y "El Orfanato"* con alguna de las métricas expuestas en UB.
2. Calcular la predicción de Marta hacia *"Rec"*.

La diferencia entre IB y UB es que en IB no se calculan los K vecinos más cercanos, sino que directamente se calcula la similitud entre productos.

2.2.2 Sistemas de recomendación basados en contenido

El proceso básico de los SR basados en contenido (BC) [12] consiste en emparejar los atributos de un perfil de usuario en el que están guardados sus preferencias e intereses, con los atributos de un producto, para así poder recomendar al usuario los nuevos artículos. Es decir, la idea básica es que un usuario elegirá productos iguales a los que ha elegido anteriormente.

Mientras que el filtrado colaborativo identifica usuarios con preferencias similares a las del usuario dado y recomienda los artículos que le han gustado, los sistemas basados en contenido recomiendan artículos similares a los que el usuario ha elegido con anterioridad.

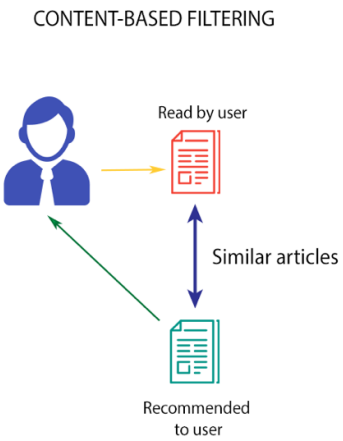

Figura 2.2. *Esquema del funcionamiento de los SR basados en contenido [10]*

A continuación, expondré algunos de los inconvenientes y las ventajas [12] de estos sistemas en comparación al filtrado colaborativo.

Ventajas:

- **INDEPENDENCIA DE LOS USUARIOS.** Solo se necesita la información de las puntuaciones de las puntuaciones dadas por el usuario activo mientras que en el FC necesitamos las puntuaciones de otros usuarios para poder utilizar el método de *vecinos cercanos* y poder hacer así una recomendación.
- **TRANSPARENCIA.** Las explicaciones para poder demostrar cómo funciona el sistema de recomendación pueden darse mirando la lista de propiedades de los artículos. Esas propiedades son indicadores para saber si la recomendación ha funcionado bien o no. Por el contrario, en el filtrado colaborativo son todo "cajas negras" porque la única explicación para la recomendación de un producto es que usuarios desconocidos con gustos similares han valorado bien ese producto.
- **NUEVOS PRODUCTOS.** Los SR basados en contenido son capaces de recomendar artículos que todavía no han sido puntuados. En el caso contrario está el

filtrado colaborativo, que necesita que un número de usuarios substancial haya puntuado el producto para poder hacer una recomendación fiable.

Desventajas:

- **ANALISIS DE CONTENIDO LIMITADO.** Las técnicas basadas en contenido tienen un número limitado de características y en algunas ocasiones este número de propiedades no son suficientes para poder distinguir aspectos de los productos que pueden ser necesarios para hacer una recomendación óptima.
- **SOBRE ESPECIALIZACION.** Este tipo de desventaja recibe el nombre de *"serendipia"* para subrayar la tendencia de los sistemas basados en contenido de recomendar con un grado limitado de novedad. Una técnica perfecta basada en contenido va a encontrar muy pocas veces una *novedad*, limitando así el rango de aplicaciones para el que va a ser válido.
- **NUEVOS USUARIOS.** Un sistema basado en contenido necesita un número elevado de puntuaciones para poder entender las preferencias de los usuarios y hacer una recomendación precisa. Por lo tanto, cuando hay pocas puntuaciones, para un usuario nuevo, no habrá buenos resultados.

Para este tipo de sistemas lo más importante es la manera en la que se representa cada producto [13], es decir, como están expuestas sus propiedades. Cuando tenemos expuestas las preferencias del usuario en base a las propiedades de cada producto lo único que hace falta es juntar las características del producto con las preferencias del usuario. Para obtener esta información se utilizan dos técnicas: **inverse document frequency** (IDF) y **term-frequency** (TF). Este tipo de técnicas se usan en productos que tiene una información textual asociada, como puede ser el nombre o la descripción de un artículo, o en documentos.

La técnica **TF** (2.1) describe la frecuencia con la que aparece una palabra en un documento. En ella se normaliza la frecuencia de que cierta palabra aparezca en el documento dividiendo esta cantidad entre el máximo de la frecuencia de aparición del resto de palabras.

$$TF(i,j) = \frac{frecuencia(i,j)}{máx\{frecuenciaotros(i,j)\}} \quad (2.1)$$

IDF (2.2) reduce el peso de las palabras que se repiten con frecuencia en el documento. De esta manera las palabras que aparezcan con menor frecuencia tendrán más peso porque las palabras muy frecuentes no son de mucha ayuda a la hora de hacer la discriminación de documentos.

$$IDF(i,j) = \log \frac{número\ total\ de\ documentos}{documentos\ en\ los\ que\ aparece\ palabra\ i} = \log \frac{N}{n(i)} \quad (2.2)$$

La combinación de ambas medidas se calcula como el producto de estas mismas.

$$TF - IDF(i,j) = TF(i,j) * IDF(i) \quad (2.3)$$

Se consideran dos usuarios *Marta* y *Jaime* y dos películas como *Terminator* y *Rec*. Los resultados de la técnica TF-IDF para este ejemplo se suponen los siguientes:

	Miedo	Acción	Suspense	Romance	Comedia
Terminator	0.35	0.7	0.2	0.02	0
Rec	0.8	0.3	0.5	0	0

Tabla 2.1*.Resultados de la técnica TF-IDF para el ejemplo propuesto*

Por último, se crea el espacio de vectores o *"Vector Space Model"* [14] que calcula la similitud entre artículos basándose en el ángulo de los vectores que se forman.

$$\cos(Terminator, Rec) = 0.6 * 0.8 + 0.4 * 0.3 + 0.2 * 0.5 = 0.7 \quad (2.4)$$

Por tanto, la similitud entre las dos películas tendrá un valor de 0.7. Este valor es suficientemente alto como para obtener la conclusión de que las películas pueden ser similares. Teniendo en cuenta los resultados de la Tabla 2.1 el espacio de vectores sería el observado en la Figura 2.3.

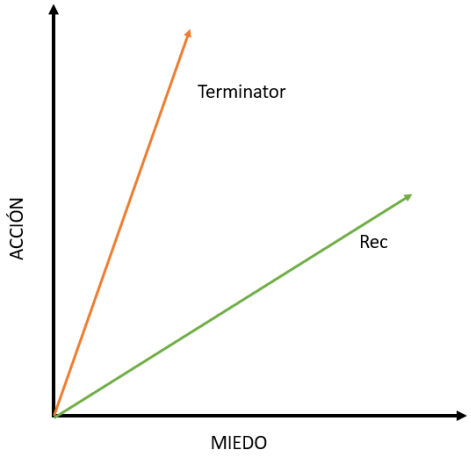

Figura 2.3. *Espacio de vectores del ejemplo para SR basados en contenido*

2.2.3 Sistemas de recomendación basados en conocimiento

Como hemos visto los dos tipos de SR anteriores tienen sus puntos fuertes para determinados casos, pero hay algunos momentos en los que ninguno de los dos anteriormente mencionados encaja.

En situaciones que hacemos una vez cada mucho tiempo, como por ejemplo comprar una casa. En este escenario un filtrado colaborativo puro no funcionaría correctamente debido al número tan pequeño de puntuaciones que habrá. Otro ejemplo es la compra de un artículo de tecnología en el que las puntuaciones que se han dado hace un cierto tiempo quizás no sean útiles ahora por el avance de la tecnología.

Los SR basados en conocimiento [15] se adaptan a cada contexto. La mayor ventaja que tienen es que no necesitan puntuaciones por lo tanto no existe el problema del arranque en frío. Hay dos tipos de sistemas: basados en restricciones *(constraint-based)* y basados en casos *(case-based)*. En ambas aproximaciones el usuario deberá especificar sus requerimientos y el sistema buscará una solución. Sin embargo, difieren en la manera de utilizar la información que se les da.

El razonamiento *case-based* resuelve problemas nuevos utilizando información de casos anteriores que han tenido un resultado satisfactorio. Primero guarda casos anteriores en memoria y para resolver los nuevos supuestos vuelve a usar experiencias similares para usuarios similares. La reutilización puede ser parcial y hay que cambiar el algoritmo o en

algunos casos se puede reutilizar por completo. Por último, se guarda este nuevo problema en la memoria para que el algoritmo aprenda.

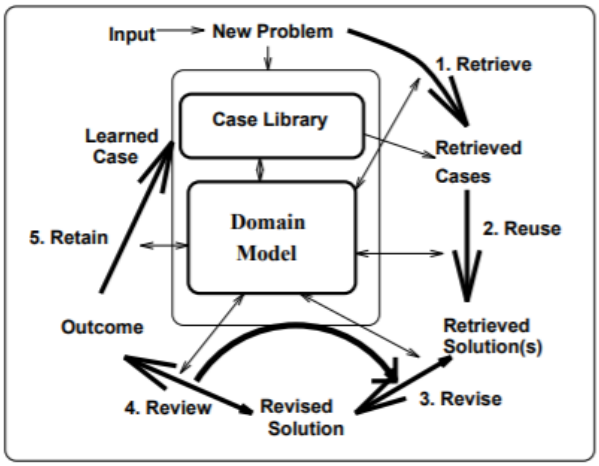

Figura 2.4. *Diagrama de flujo recomendación case-based [16]*

El funcionamiento de un SR *constraint-based* se basa en los filtros (preferencias) que quiera poner el usuario. En primer lugar, el usuario especifica sus preferencias iniciales que pueden ser comunes para todos los usuarios o personalizadas. Esto se suele hacer con una serie de preguntas o marcando opciones. En segundo lugar, se introduce la información dentro del algoritmo y se presentan una serie de resultados. Finalmente, al revisar la información que el sistema de recomendación le ha dado, el usuario tiene la opción de buscar soluciones alternativas porque el resultado no ha sido satisfactorio, o hacer una búsqueda más selectiva con sus gustos para obtener unos resultados más personalizados.

La implementación práctica de este tipo de sistemas no es sencilla puesto que tiene que adaptarse a cada usuario y cada contexto en concreto. Los modelos de predicción que se suelen utilizar en este tipo de sistemas son los arboles de decisión ya que mediante preguntas se puede ir guiando al usuario hacia un resultado final.

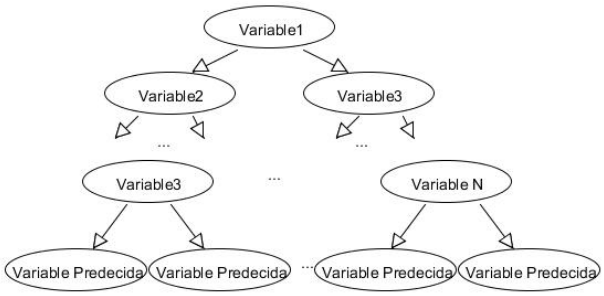

Figura 2.5. *Árbol de decisión [17]*

Algunos de los problemas que se tratan de evitar son los resultados por defecto, los conjuntos vacíos y la lentitud computacional.

2.2.4 Sistemas de recomendación híbridos

Todos los tipos de SR descritos hasta ahora tienen que lidiar con una serie de desventajas como la dispersión de los datos, el arranque en frío, la dificultad para obtener características de los productos… Debido a la necesidad de un sistema "total" que combinara técnicas dos o más técnicas de recomendación surgen los SR híbridos [18,19].

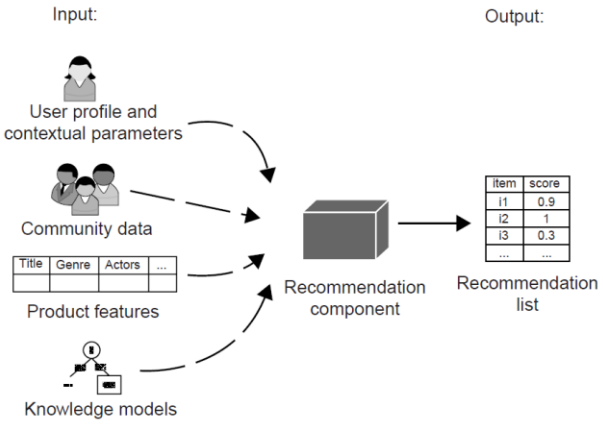

Figura 2.6*. Diagrama de flujo sistema de recomendación híbrido [18]*

En esta imagen se puede observar como un sistema híbrido recopila el perfil del usuario y características de los productos necesarios cuando utilizamos sistemas basados en contenido, una comunidad de información necesario en el filtrado colaborativo y modelos de conocimiento necesarios en los sistemas basados en conocimiento, para finalmente obtener una lista de recomendaciones.

Como se muestra en la Figura 2.4 para un sistema híbrido se necesitan las técnicas tanto de filtrado colaborativo como basado en contenido y basado en conocimiento. El filtrado colaborativo asume que hay una serie de clusters formados por usuarios que tienen preferencias similares. La función del FC entonces será buscar parejas similares y hacer recomendaciones a partir de los productos favoritos. Por otro lado, la técnica basada en contenido hace una función muy parecida al FC puesto que recomienda productos a usuarios que han tenido unos gustos similares en el pasado. Finalmente, los basados en conocimiento introducen la personalización del sistema, como hemos visto antes en el Capítulo 2.2.3 esta personalización puede venir dada por restricciones elegidas por el usuario o por casos anteriores.

La elección del sistema final que vamos a utilizar depende de la información de entrada de la que dispongamos. Existen siete métodos de hibridación [19] que se procederán a explicar en la siguiente tabla.

Método	**Descripción**
Weighted	Las puntuaciones de varios SR se combinan para obtener el resultado
Switching	Se cambia de SR dependiendo de los resultados que se obtengan
Mixed	Se presentan de manera conjunta los resultados de varios SR
Feature Combination	Propiedades de varios SR se lanzan juntos a un mismo algoritmo
Cascade	Un SR refina los resultados que ha obtenido el anterior
Feature augmentation	La salida de un SR se utiliza como entrada de otro
Meta-level	El modelo que ha aprendido un SR se utiliza como entrada para otro

Tabla 2.2. *Métodos de hibridación [19]*

3 ANALISIS DE LA BASE DE DATOS

En este capítulo se llevará a cabo un análisis de la base de datos utilizada para la solución del sistema de recomendación.
La base de datos utilizada es 3 Million Instacart Orders**.** Es una base de datos libre de licencia, extraída de la página web de Kaggle, cuya competición se encuentra cerrada desde hace meses. La competición consistía en predecir qué productos estarán en el próximo pedido de cada usuario. La empresa que proporciona la BBDD es **Instacart**, desde cuya aplicación se puede hacer pedidos de alimentos.

Hay dos ficheros principales que son los que usare en el sistema de recomendación. Uno de ellos contiene toda la información acerca de cada pedido por parte de cada usuario. El segundo de ellos contiene la información relativa a cada uno de los productos.

El análisis de esta base de datos ha sido realizado tanto para un sistema de recomendación con filtrado colaborativo e información implícita como para un sistema de recomendación basado en contenido. Se desarrollarán dos análisis puesto que cada tipo de SR utiliza una información distinta. El filtrado colaborativo recomendará productos que usuarios similares han elegido en el pasado y para ello será necesario un análisis de las puntuaciones. El sistema basado en contenido recomendará productos similares a los que ese usuario ha elegido en el pasado y será necesario un análisis de las relaciones entre artículos.

3.1 Estructura inicial

La BBDD consta de 3.000.000 de pedidos por parte de 200.000 usuarios. Para cada usuario se proporciona información sobre entre 4 y 10 pedidos, así como la semana y la hora del día en la que el pedido fue hecho y la frecuencia con la que se adquiere cada producto. Como punto de partido tenemos dos ficheros distintos.

El primero de los ficheros contiene la información de los 3 millones de pedidos. En la siguiente tabla se detallan los datos que contiene este primer fichero y su descripción.

order_id	user_id	eval_set	order_number	order_dow	order_hour_of_day	days_since_prior_order
1	112108	train	4	4	10	9.0
36	79431	train	23	6	18	30.0
38	42756	train	6	6	16	24.0
96	17227	train	7	6	20	30.0
98	56463	train	41	3	8	14.0

Figura 3.1*. Estructura primer fichero*

- ***Order id****:* identificador de cada pedido.
- ***User id:*** identificador del usuario que ha realizado el pedido.
- ***Eval set****:* conjunto al que pertenece cada dato (train, test o prior).
- ***Order number****:* número del pedido dentro de cada usuario.
- ***Order dow:*** día de la semana en el que se ha hecho el pedido. Tiene un rango del 1 al 6.
- ***Order hour of day:*** hora del día en la cual se ha efectuado el pedido. Formato 24 horas.
- ***Days since prior order:*** variable que indica la frecuencia con la que se hizo el anterior pedido. Se indica en días, por tanto, la variable tiene un rango de 1-30.

Este primer archivo solo nos da información sobre los pedidos que realiza cada usuario, pero para tener datos sobre los productos que contiene cada pedido se necesitará el segundo de los ficheros. La estructura de este segundo fichero se expone en la siguiente figura.

order_id	product_id	add_to_cart_order	reordered
1	49302	1	1
1	11109	2	1
1	10246	3	0
1	49683	4	0
1	43633	5	1

Figura 3.2. *Estructura segundo fichero*

- ***Order id:*** identificador del pedido.
- ***Product id:*** identificador del producto.
- ***Add to cart order:*** orden en el que se introducen los productos en cada pedido.
- ***Reordered:*** variable con valor 0 ó 1 que indica si el producto ha vuelto a ser pedido.

3.2 Análisis de puntuaciones para el sistema de filtrado colaborativo

Debido a que el filtrado colaborativo se basa en recomendar artículos que han sido elegidos por usuarios similares, debemos hacer un análisis exhaustivo del comportamiento de cada usuario para obtener puntuaciones implícitas. El primero de los archivos que se va a utilizar contiene un total de 3.421.083 de pedidos. Existen tres tipos de conjuntos de datos distintos: entrenamiento, test y prior. En la siguiente gráfica se muestra el número de pedidos que contiene cada conjunto.

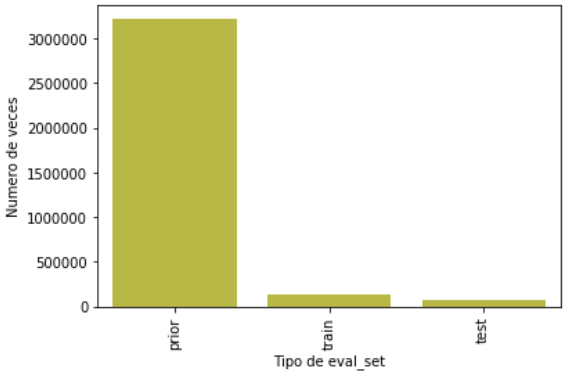

Figura 3.3*.Conjunto de datos*

Para seleccionar el conjunto de datos que se va a utilizar hay que tener en cuenta varios factores. El más importante, es que solo existe fichero con información sobre los productos que contiene cada pedido (Figura 3.2) para el conjunto de entrenamiento y el de prior. Por ello, el conjunto de test queda descartado para su utilización en este SR. Para tomar la decisión final sobre cuál de los conjuntos (entrenamiento o prior) se va a utilizar se realiza un análisis de los estos dos conjuntos.

3.2.1 Análisis de los pedidos por parte de los usuarios

Al examinar el primero de los archivos restringiendo los datos (solo con los elementos que pertenecen al conjunto de *"train"*), podemos obtener varias conclusiones.

Analizando el número de pedidos que hace cada usuario obtenemos la Figura 3.4 en la cual el eje de abscisas corresponde al recuento de usuarios y el eje de ordenadas corresponde al número de pedidos que hace cada usuario. De esta manera, podemos ver que hay un alto número de usuarios, en torno a 15.000, que han hecho cuatro pedidos y en el lado opuesto hay un número muy pequeño de usuarios que han realizado cien pedidos. Esta figura tiene una pendiente decreciente que indica que la mayoría de los usuarios tiene muy pocos pedidos registrados en la aplicación, pero ya que hay un gran número de usuarios esto no debería suponer un problema. La pendiente de esta gráfica es la que se espera en un SR, por tanto, el comportamiento de nuestro sistema es el esperado. Esto demuestra que la matriz final de puntuaciones que se obtendrá en apartados posteriores será dispersa debido a los pocos datos de los que se dispone.

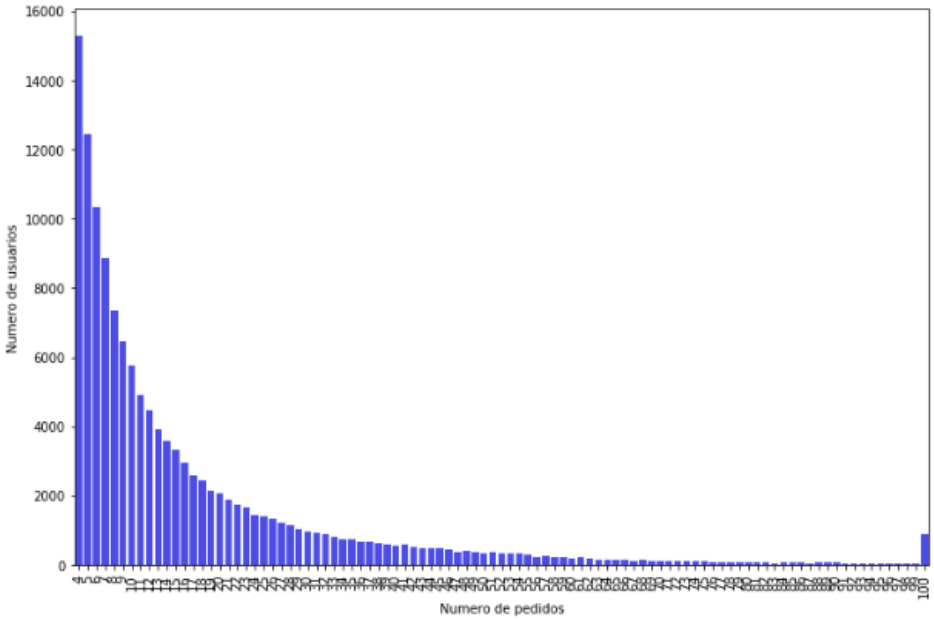

Figura 3.4. *Cantidad de pedidos por usuario*

En cuanto a la distribución de los pedidos durante los días de la semana, podemos apreciar en la Figura 3.5 que hay un claro incremento de pedidos en el primer día de la semana mientras que en la mitad de la semana se registra la mayor bajada.

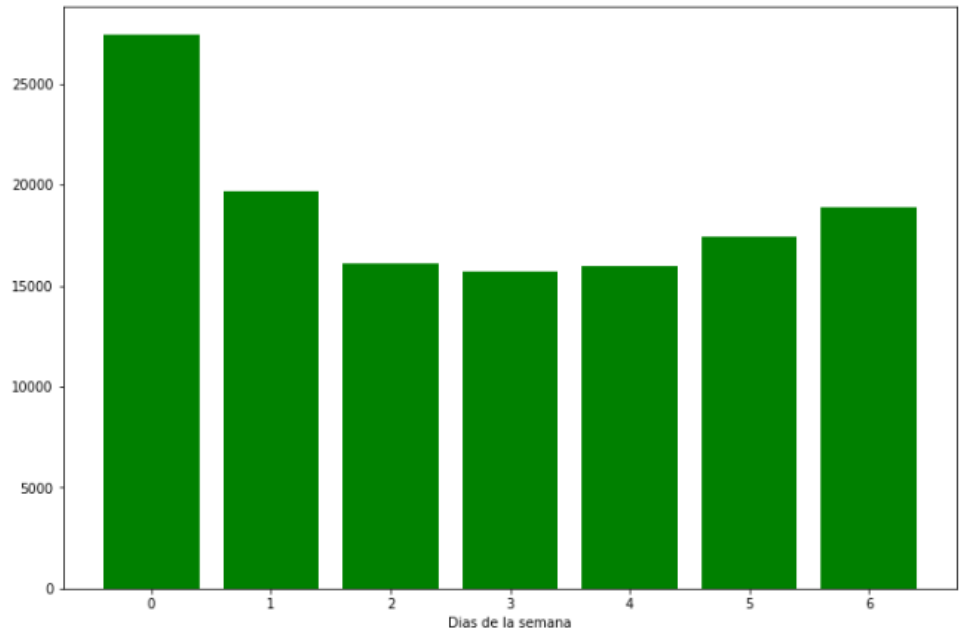

Figura 3.5. *Pedidos por días de la semana*

En el siguiente mapa de calor (Figura 3.6) examinamos la distribución de los pedidos teniendo en cuenta el día de la semana y la hora del día. Podemos observar que los puntos "más calientes" están situados en los primeros días de la semana y en los dos últimos. Teniendo en cuenta la hora del día, se aprecia una franja de mayor cantidad de pedidos entre las 10 de la mañana y las 4 de la tarde.

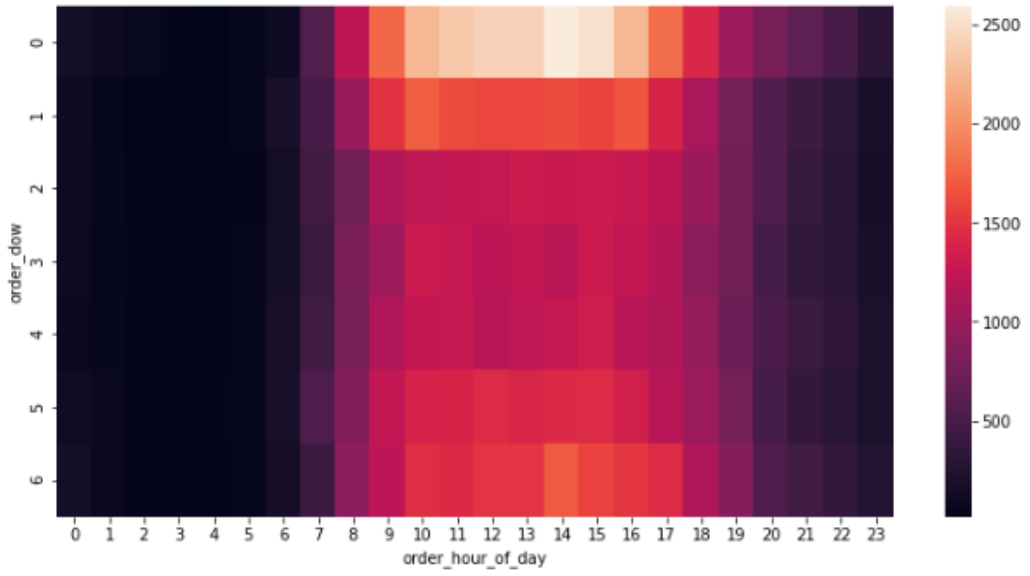

Figura 3.6. *Mapa de calor frecuencia de pedidos en el día de la semana frente a horas del día*

Por último, es importante cuál es la frecuencia con la que un usuario suele volver a hacer un pedido, ya que como veremos en el siguiente capítulo, esto influye en las puntuaciones que cada usuario le da a cada producto.

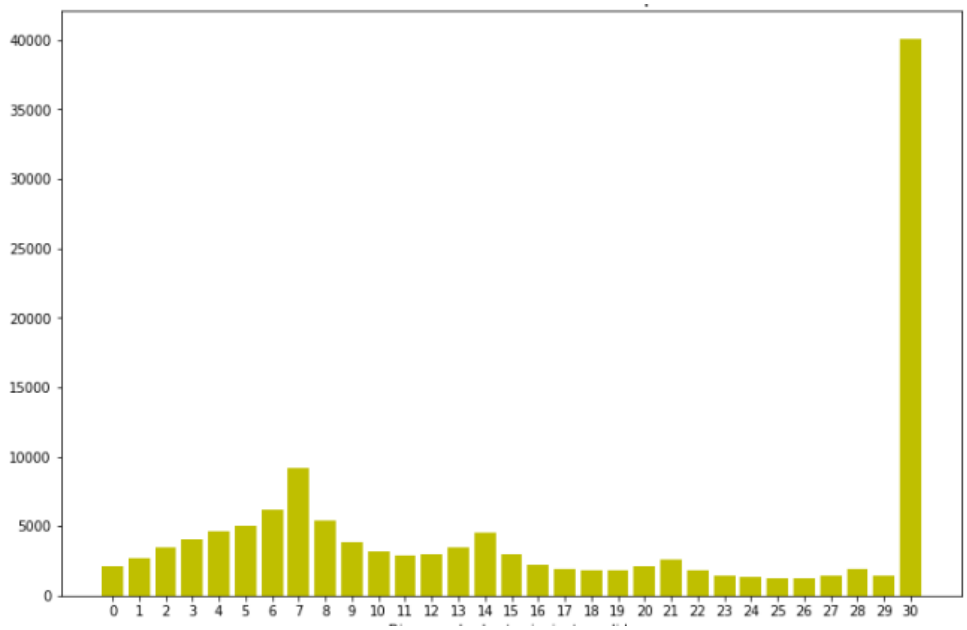

Figura 3.7*. Frecuencia con la que un usuario vuelve a hacer un pedido en el conjunto de entrenamiento*

En ella se observa como la mayoría de los usuarios esperan hasta los 30 días para volver a efectuar cada pedido. Si se analiza la misma Figura 3.7 para el conjunto prior se obtiene la siguiente figura.

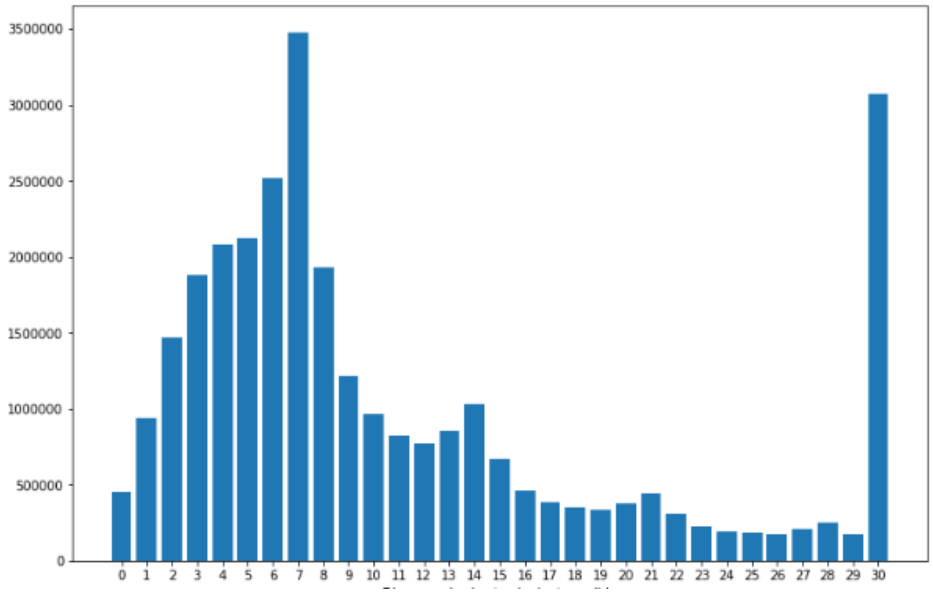

Figura 3.8. *Frecuencia con la que un usuario vuelve a hacer un pedido en el conjunto de prior*

Una vez analizada la frecuencia con la que se realizan los pedidos en ambos conjuntos (train y prior) se puede observar que hay más variedad de valores para el conjunto de prior ya que en el de entrenamiento la mayoría de los pedidos se realizan cada 30 días. Para obtener mejores resultados es necesario tener mayor cantidad de información.

Analizando el número de usuarios que forma cada conjunto de datos y el número de filas que contiene cada conjunto, se obtiene la siguiente información sobre ambos conjuntos:

	Número de usuarios	**Número total de elementos**
Entrenamiento (Train)	131.209	1.384.617
Prior	206.209	30.256.421

Tabla 3.1. *Número de elementos que pertenece a cada conjunto*

Dado que hay mayor cantidad de datos en el conjunto prior, se selecciona éste para construir el SR.

Cada pedido consta de una serie de productos y mediante los datos en la columna *"reordered"* podemos saber cuándo un artículo ha vuelto a ser pedido. En el caso de que el valor de esta columna para un cierto artículo sea cero se interpretará como que el usuario no ha vuelto a comprar ese artículo porque no le ha gustado.

Se debe fusionar el fichero con los datos de los pedidos con el fichero con los datos de los productos que pertenecen a cada pedido. Analizando la columna *"reordered"* se puede saber cuál es el porcentaje de productos que tienen esta variable a cero. Este valor tendrá un significado importante en la obtención de las puntuaciones a los productos, que se verá en apartados posteriores.

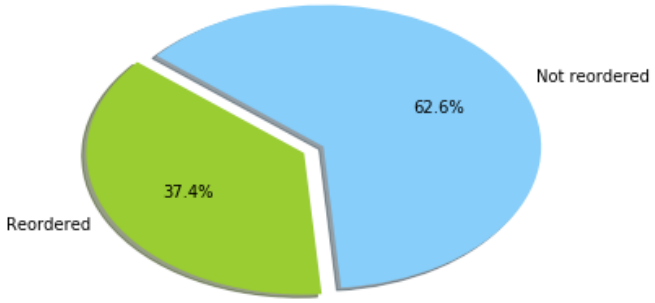

Figura 3.9. *Porcentaje de productos que tienen valor nulo en el conjunto prior*

Con relación a las veces que ha vuelto a ser pedido un producto, un estudio importante, es la segregación por departamentos. Para analizar este tema en concreto se ha diseñado la Figura 3.8 que contiene la información de la cantidad de veces que se reordenan los artículos, pero dividido en departamentos. En el eje de abscisas se encuentra la ratio y en el eje de ordenadas se encuentran los distintos departamentos.

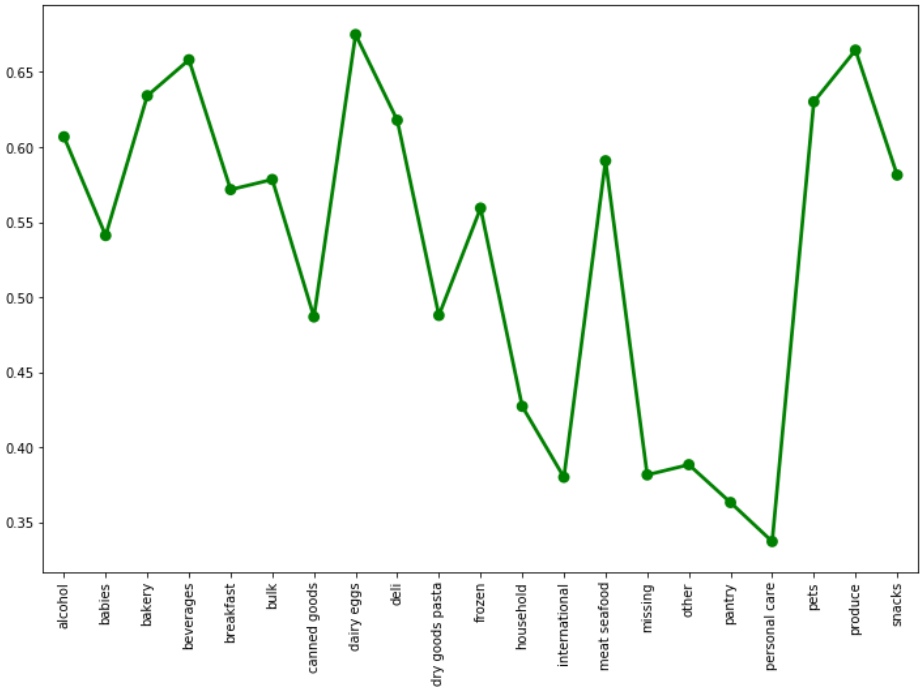

Figura 3.10. *Repetición de ordenes con respecto a cada departamento*

Al analizar la Figura 3.10 se observa que los departamentos con mayor número de popularidad para los usuarios van a ser los de huevos, comida ultra procesada y bebidas. Por el contrario, el departamento con la ratio más baja es el de higiene personal.

3.3 Base de datos para el sistema de recomendación basado en contenido

Para este tipo de SR se necesita obtener las relaciones entre los productos. Para ello se usarán las características de cada producto: pasillo y departamento al que pertenece cada artículo. Para tener una visión más gráfica de cuantos artículos hay en cada pasillo he realizado la gráfica que se muestra en la Figura 3.11 en la que se sitúan todos los distintos pasillos del supermercado en el eje de ordenadas y el recuento total de artículos en cada pasillo en el eje de abscisas.

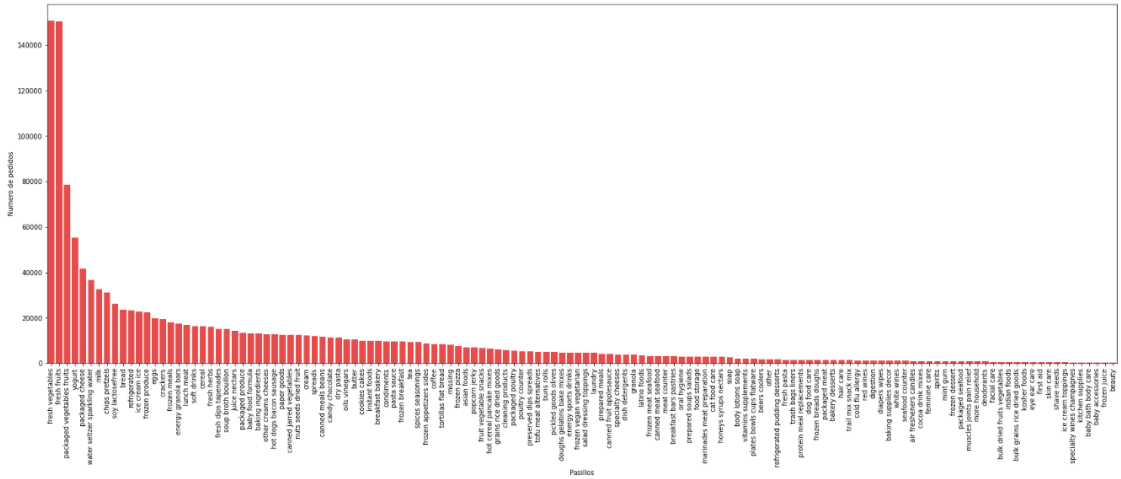

Figura 3.11. *Recuento del total de productos por pasillo. En el eje de abscisas se encuentra el recuento de productos en cada pasillo, en el de ordenadas el nombre de cada pasillo*

Si ampliáramos el gráfico podríamos ver que los dos pasillos con más productos son el de verduras frescas y frutas frescas, mientras que los que menos artículos tienen son la bollería y la fruta congelada. De esta manera podemos hacernos una idea de cómo están distribuidos los productos por pasillos.

De la misma manera podemos obtener información sobre la distribución de los productos por departamentos. Dentro de nuestra base de datos hay veintiún departamentos, en los cuales los artículos están distribuidos de la siguiente manera:

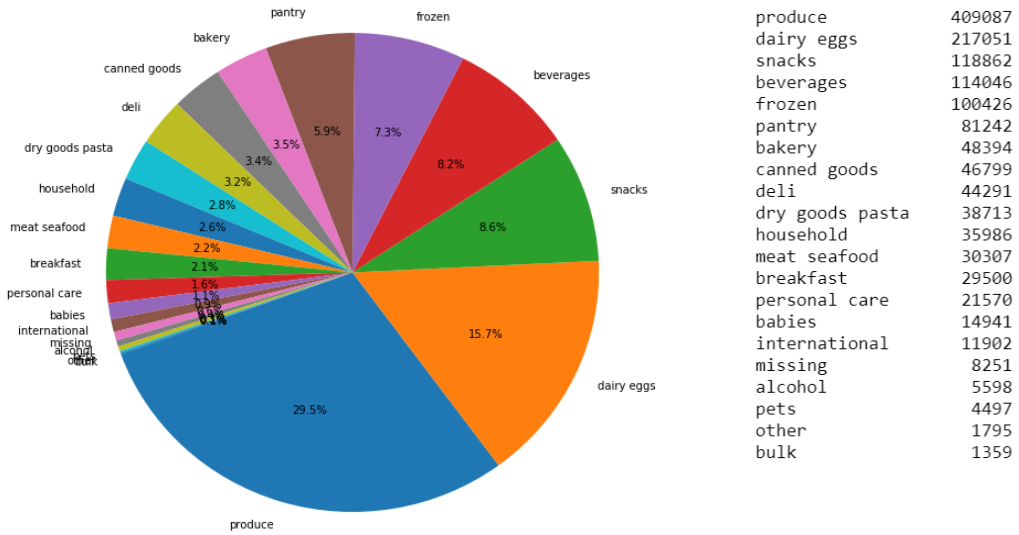

Figura 3.12. *Distribución de productos por departamentos*

Se puede observar que el departamento que más artículos contiene, con un 29,5 %, es el de comida ultra procesada. Esto es un reflejo de la realidad puesto que, según una investigación realizada por Foodwatch Holanda [20], el 70 % de los productos alimenticios de un supermercado son ultra procesados.

El fin del SR basado en contenido es encontrar productos similares a los que el usuario ha comprado y recomendárselos. En este proyecto los productos similares serán los que estén en el mismo departamento que un producto que el usuario haya comprado y para mayor similitud el top-5 de productos recomendados estará formado por productos que se sitúen en el mismo pasillo que este producto. En caso de no existir ningún artículo en el mismo pasillo, se buscarán pasillos cercanos.

3.4 Separación de los conjuntos de entrenamiento, test y validación

En muchas ocasiones los datos de entrenamiento del SR no son capaces de generalizar el modelo [21] porque le damos muy poca información, este problema se denomina *underfitting*. El otro problema que puede surgir es que le demos demasiada información al algoritmo y tampoco sea capaz de generalizar puesto que todo lo que no se ajuste a todas las características que hemos puesto será descartado, esto se denomina *sobreajuste*

(overfitting). Para evitar que estos dos problemas hagan que nuestro SR no funcione correctamente se subdivide el conjunto total de datos [22].

El subconjunto de **entrenamiento** se utiliza para entrenar los datos que le damos al sistema de recomendación, por tanto, son datos "visibles" por el modelo. Los resultados obtenidos con este subconjunto no deben tenerse en cuenta puesto que no representan el rendimiento real.

El subconjunto de **validación** se utiliza para probar modelos que se han ajustado con los datos de entrenamiento y para ajustar parámetros como el número de iteraciones del algoritmo o los parámetros de regularización de una regresión. Los datos no son visibles por el modelo con el de obtener un modelo insesgado.

El subconjunto de **test** son datos no visibles por el modelo, con ellos se mide el rendimiento real del sistema. Con este conjunto se toman las medidas de precisión como el Mean Average Precision (mAP), (Normalized Discount Cumulative Gain) nDCG o F1-score.

El objetivo es maximizar el conjunto de entrenamiento para poder tener más datos y ajustar mejor el modelo, pero también se quiere maximizar el conjunto de test para poder obtener mejores resultados al tener más información sobre la que medir el sistema. Si se incrementa el número de datos de entrenamiento, se reducirán los datos de test, por ello se utiliza el proceso de **validación cruzada** [23]**.** Dentro de la validación cruzada existen varios tipos, pero el más utilizado es el de K iteraciones.
Consiste en dividir el conjunto de entrenamiento en tantas partes como indique la K, después se coge una de las partes para test y el resto para entrenar. El paso siguiente es entrenar el algoritmo con los datos de entrenamiento y ver su rendimiento con los de test. Una vez hecho esto, la siguiente partición pasará a ser datos de test y el resto de entrenamiento añadiendo la parte que antes era de test al conjunto de entrenamiento. Este proceso se hará durante K iteraciones. Cuando bucle finalice se calculará el error en test medio para saber cuál será el error final.
Este proceso se utiliza también para ajustar los parámetros de regularización y las iteraciones dentro de cada algoritmo cogiendo siempre el valor que minimice el error en test.

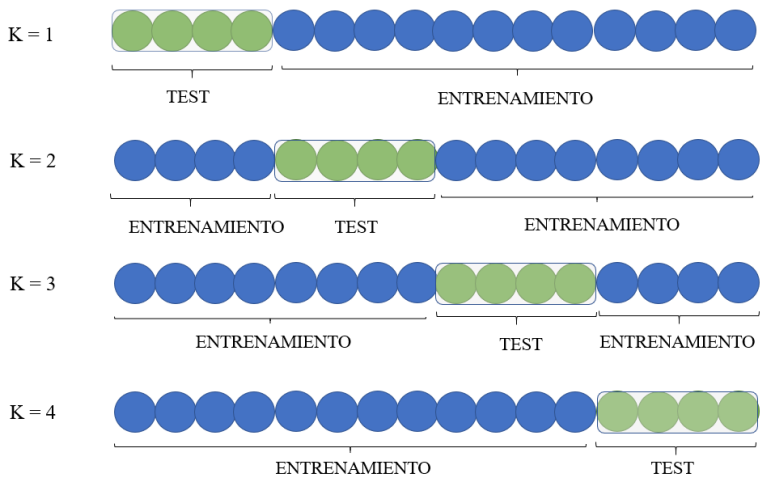

Figura 3.13. *Ejemplo para la explicación de la validación cruzada con 4 iteraciones*

En los sistemas de machine learning, el conjunto de entrenamiento corresponde al 80 % de los datos, mientras que el de test y el de validación se llevan el otro 20 %.
Para el caso concreto de este proyecto el conjunto de entrenamiento estará compuesto por 4.972 productos y 7.000 usuarios, el de test por 4.962 productos y 7.000 usuarios y el de validación por 4.957 productos y 7.000 usuarios.

Para los SR con filtrado colaborativo es muy importante el nivel de dispersión de los datos. El problema del arranque en frío es algo muy presente en cualquier filtrado colaborativo. Por este motivo se ha de comprobar que los datos no sean demasiado dispersos para que los resultados sean buenos. Como se va a trabajar con matrices muy dispersas, un nivel de dispersión óptimo para estas matrices será desde el 97 % al 99%.

3.5 Obtención de la matriz de puntuaciones

Como se ha visto en el Capítulo 3.1 el archivo que vamos a utilizar para crear la matriz de puntuaciones es el de la Figura 3.1 y la Figura 3.2. Estas puntuaciones serán implícitas, por tanto, hay que analizar el comportamiento del usuario para obtenerlas. Tras formar una sola base de datos después de juntar ambos ficheros podemos observar que hay dos columnas que pueden darnos los datos necesarios para generar estas puntuaciones implícitas. Una de las columnas es la de "*reordered"* que indica con un 1 ó un 0 si el producto ha vuelto a ser adquirido. La otra columna es la de "*days since*

prior order", que indica la frecuencia con la que ese producto ha sido pedido. Esta columna es lo que en un SR de música o de películas sería la columna que indica el número de veces que se ha visionado la película o el número de veces que se ha escuchado una canción.

Para darle una calificación nula a los productos que nunca han sido pedidos y un valor no nulo a los productos cuyo valor en la columna reordered es uno, multiplicaremos la primera de las columnas descritas por la segunda.

$$Rating = \frac{reordered}{days\ since\ prior\ order} \tag{3.1}$$

El problema de realizar esta operación para calcular las puntuaciones es que cuanto más alto sea el valor de los días que pasan hasta que se vuelve a adquirir un producto más alta será la puntuación que se le asigna al producto. El efecto que se quiere conseguir es asignarle una puntuación más alta a los productos que se compran con más regularidad. Por tanto, se realizará la inversa de cada puntuación obtenida con la ecuación 3.1. Para poder determinar si la columna de puntuaciones está bien creada y tiene los valores que necesita el sistema para obtener buenos resultados, se crea un diagrama de caja en la Figura 3.14 que muestra los valores mínimo y máximo dentro de esta columna y la distribución del resto de valores.

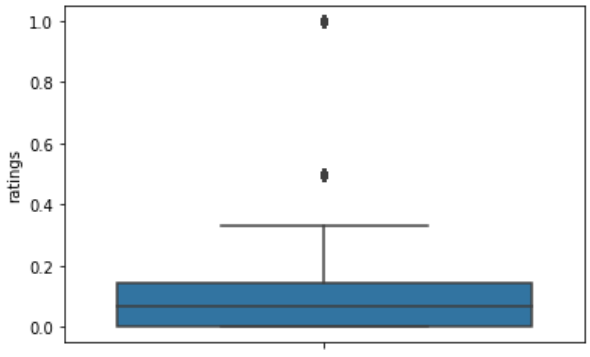

Figura 3.14. *Diagrama de caja de la columna de puntuaciones inicial*

Observamos como la mayoría de los valores se encuentra entre 0 y 0.2, el valor mínimo es el 0 y el valor máximo es 1. La mediana de los datos está en torno al 0.08.

3.6 Solución al problema de la dispersión de datos

En una primera instancia el sistema de recomendación fue implementado con la totalidad de los datos, los cuales tenían un nivel de dispersión del 99,9%. Debido a que este nivel no es aceptable puesto que el resultado del algoritmo Alternating Least Squares (ALS) sería una matriz con la mayoría de las posiciones nulas, se decidió desarrollar una solución.

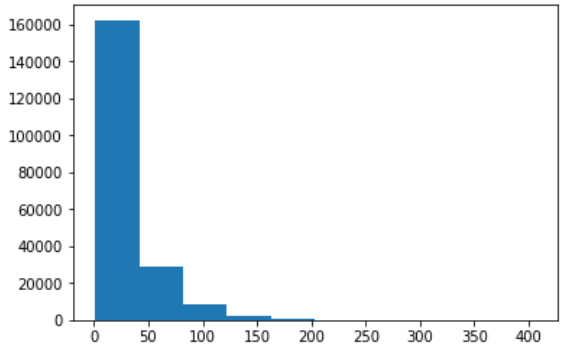

Figura 3.15. *Histograma del número de apariciones de cada usuario en el conjunto inicial de datos*

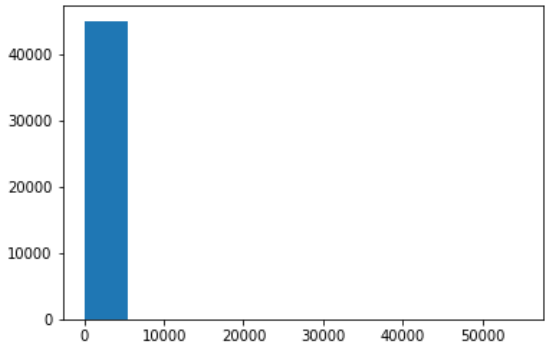

Figura 3.16. *Histograma del número de apariciones de cada producto en el conjunto inicial de datos*

La solución más simple y con mayor eficiencia es filtrar el número de usuarios y productos que se introducen en el algoritmo. Los datos van a seguir siendo dispersos ya que un gran número de usuarios puntúa en muy pocas ocasiones ciertos artículos. La diferencia va a residir en que en esta ocasión vamos a coger solamente los usuarios que más artículos hayan puntuado y los productos más puntuados. De esta manera si imaginamos una gráfica con pendiente decreciente que indique el número de artículos calificado por cada usuario solo cogeremos la cúspide de esa gráfica para así obtener una matriz mucho menos dispersa como resultado.

Dado que disponemos del identificador de cada usuario y de cada producto este proceso se vuelve más simple ya que solo tenemos que observar el número de repeticiones de cada usuario dentro de nuestra matriz y escoger los que tengan un mayor valor, y hacer el mismo proceso con los artículos. Solo se introducirán en el algoritmo los primeros 8.000 usuarios con más productos puntuados y los 5.000 productos más puntuados. De esta manera, la matriz de entrenamiento tiene un nivel de dispersión del 98,3 %.

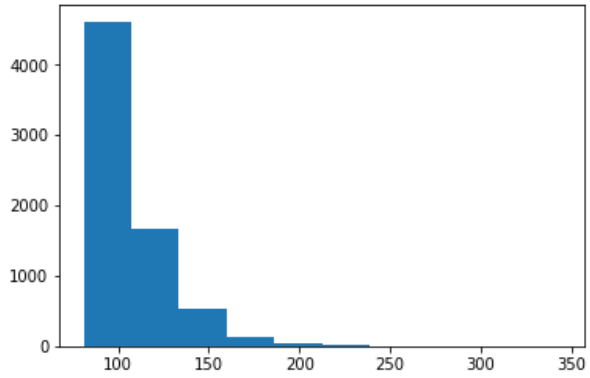

Figura 3.17. *Histograma de la distribución del número de apariciones de los usuarios en el conjunto de datos filtrado*

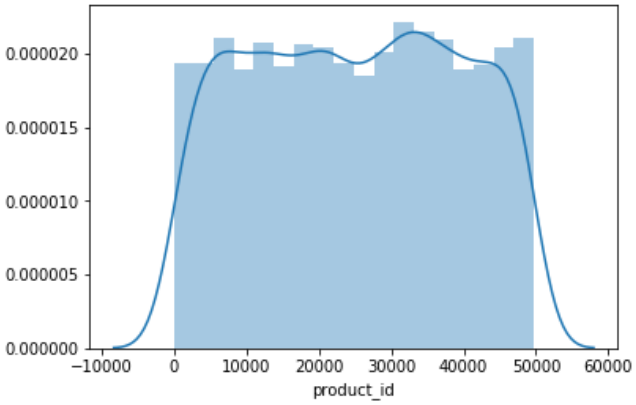

Figura 3.18*. Histograma de la distribución del número de apariciones de los productos en el conjunto de datos filtrado*

3.7 Análisis del conjunto de entrenamiento

Una parte importante del SR es tener la mayor información posible sobre el conjunto de datos que se va a entrenar para tener una visión previa de los resultados que se podrían obtener. Como se explicará en el capítulo 5, los resultados de las métricas de estos sistemas dependen en gran medida de los datos iniciales que se tienen. Si estos datos

son muy simples y nuestra BBDD está extremadamente preparada para introducirla en cualquier sistema, los resultados deberán ser muy buenos para poder decir que nuestra implementación es correcta. Sin embargo, en un caso real los datos no van a ser tan perfectos, y por ello los resultados no serán tan buenos como en el caso ideal, pero serán óptimos para nuestro caso en concreto. En las siguiente gráficas se muestran tanto la distribución de usuarios como la de productos para el conjunto de entrenamiento.

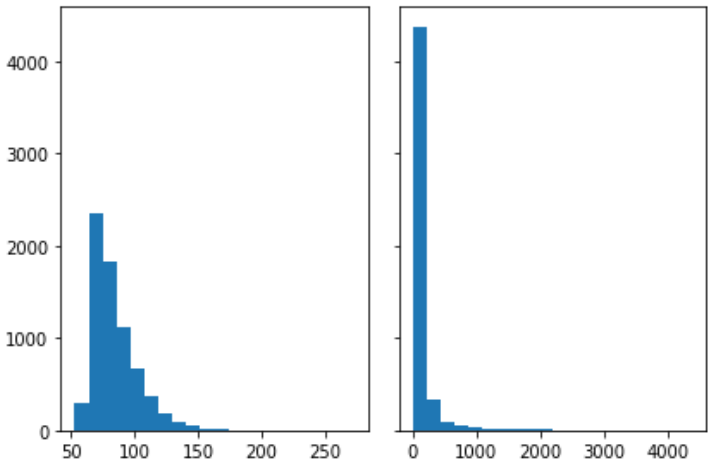

***Figura 3.19**. Histogramas de la distribución del número de usuarios y productos en el conjunto de entrenamiento*

En cuanto a las puntuaciones del conjunto de entrenamiento, se demuestra mediante el diagrama de caja de la Figura 3.20 que la mayoría de los productos tienen puntuaciones bajas por parte de los usuarios. El valor mínimo de las puntuaciones del conjunto de entrenamiento es 0.0333, el máximo 1.0, la mediana es 0.1666 y la media es 0.24. Existen valores atípicos, es decir, menos frecuentes, como son el 1.0 y el 0.5 puesto que esos valores corresponden a los productos que se han vuelto a comprar al día siguiente o a los dos días.

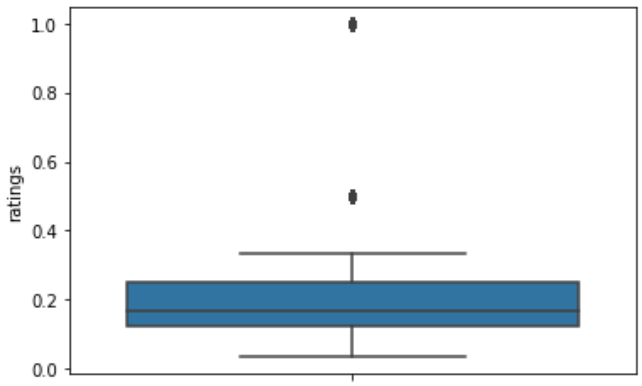

Figura 3.20*. Diagrama de caja con la distribución de las puntuaciones en el conjunto de entrenamiento*

Para poder saber qué usuarios son los más activos, es decir, los que compran productos con más frecuencia, se realiza la media de las puntuaciones que cada usuario da a los productos. En el siguiente diagrama de caja se muestra la distribución de estas medias, y se observa que hay dos usuarios con valores más altos que el resto. Estos dos usuarios son 54485 y 172806 con una media de puntuaciones de 0.95 y 0.94 respectivamente.

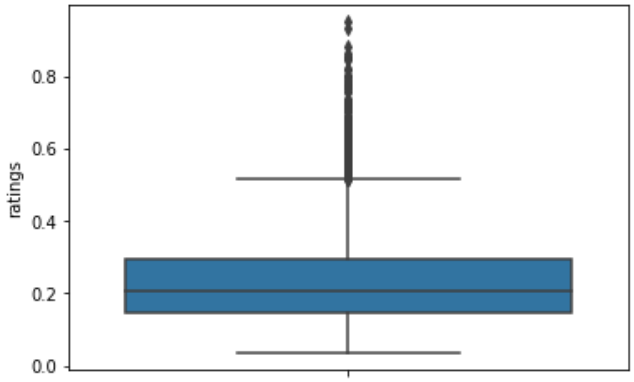

Figura 3.21. *Diagrama de caja con la distribución de la media de puntuaciones dada por cada usuario en el conjunto de entrenamiento*

4 ELEMENTOS DE UN SISTEMA DE RECOMENDACIÓN CON FILTRADO COLABORATIVO

En este capítulo se explicarán todos los elementos que forman parte de un SR con FC. Como punto de partida es necesario explicar la importancia que tienen las puntuaciones en este tipo de sistemas. Para finalizar el capítulo se describirán algunas de las métricas que se utilizan para medir la precisión del sistema de recomendación.

4.1 Tipos de puntuaciones para filtrado colaborativo [9]

Cualquier sistema de recomendación con FC necesita información por parte del usuario. Lo que llamamos actualmente puntuaciones. Existen dos tipos de información que el usuario puede dar al sistema de recomendación. En los principios del FC solo se utilizaba la información **explícita**, pero dado que surgieron una serie de inconvenientes, cada vez es más utilizada la información implícita.

Las puntuaciones explícitas son las valoraciones que un usuario da al comprar cualquier producto, como, por ejemplo, evaluar con estrellas un producto, una película o un libro. El principal problema de este tipo de puntuaciones es que son muy difíciles de conseguir. Esto es debido a que los usuarios no quieren valorar los artículos que compran, en muchas ocasiones porque les supone una pérdida de tiempo. Esta dificultad para conseguir calificaciones hace que la matriz que se forme de usuarios y artículos tienda a ser dispersa.

Una de las ventajas de los métodos de factorización matricial que veremos más adelante, es que podemos darle al sistema de recomendación más información cuando no hay suficientes puntuaciones explícitas. Toda esta información adicional forma la realimentación **implícita**.

Se obtienen indirectamente del comportamiento del usuario en cualquier web. Cuando un usuario compra un producto, supone una puntuación positiva para ese artículo puesto que si lo ha adquirido es porque era del agrado del consumidor. Otro ejemplo podría ser la frecuencia con la que un usuario visualiza un determinado artículo en internet o la

frecuencia con la que compra cierto artículo. Si las frecuencias son muy elevadas, significa que ese producto tendrá una puntuación elevada para el usuario.

Este tipo de puntuaciones no son del todo fiables y pueden llevar a error en algunos momentos. Esta tendencia al error suele pasar con las puntuaciones nulas ya que el SR interpreta una calificación nula como algo que no ha gustado. Sin embargo, esto no tiene por qué ser del todo cierto, ya que puede deberse al desconocimiento del producto. En el Capítulo 4.2.2 veremos cómo solucionar este inconveniente.

4.2 Algoritmos para el sistema de recomendación con filtrado colaborativo

Dos de las áreas del filtrado colaborativo son los métodos de vecinos y los modelos de factores latentes. Los métodos de vecinos cercanos se centran en las relaciones entre productos o entre usuarios. Sin embargo, este método no es tan efectivo en el FC puesto que la matriz será dispersa y tendrá muchos elementos vacíos. Por esta razón será mucho más complicado encontrar vecinos cercanos ya que la mayoría serán cero. Estos métodos de vecinos cercanos son usados en los modelos que se describen en el apartado 2: basados en usuarios y basados en productos. Como ya se ha explicado con anterioridad, el FC introduce el *"cold-start problem"*, lo que implica una dispersión de los datos. Para solucionar este problema se introduce el modelo de factores latentes para averiguar la similitud entre usuarios y productos.

4.2.1 Métodos de vecinos cercanos [24]

Los métodos de vecinos cercanos se basan en el algoritmo KNN (K-nearest neighbours). La finalidad de este algoritmo es encontrar los elementos más cercanos a un elemento activo como se ha explicado en el capítulo 2.2.2. Para poder encontrar estos elementos cercanos se establecen unas medidas de similitud basadas en la distancia entre dos puntos. Una de estas medidas es la similitud del coseno *(cosine similarity)* que calcula la similitud de dos elementos con el coseno del ángulo entre los dos vectores de los dos elementos.

$$\cos(item_1, item_2) = \frac{\sum_{user} rating_{user,item_1} * rating_{user,item_2}}{\sqrt{\sum_{user}(rating_{user,item_1})^2} * \sqrt{\sum_{user}(rating_{user,item_2})^2}} \quad (4.1)$$

Otra de las medidas que se utiliza es la correlación de Pearson, en la que se usan las puntuaciones medias que han sido dadas a cada producto o las puntuaciones medias que cada usuario ha dado a todos los productos que ha puntuado.

$$sim(x, y') = \frac{\sum_{s \in s_{xy}} (r_{x,s} - \overline{r_x})(r_{y,s} - \overline{r_y})}{\sqrt{\sum_{s \in s_{xy}} (r_{x,s} - \overline{r_x})^2} \sqrt{\sum_{s \in s_{xy}} (r_{y,s} - \overline{r_y})^2}} \quad (4.2)$$

$s \rightarrow usuario\ que\ ha\ puntuado\ tanto\ el\ item_1\ como\ el\ item_2$

$x \rightarrow item1 \qquad y \rightarrow item2$

$s_{xy} \rightarrow conjunto\ de\ usuarios\ que\ han\ puntuado\ ambos\ productos$

Una vez se ha calculado la similitud entre productos se utilizan medidas de predicción como la expuesta. En esta expresión se utiliza la puntuación media de los dos productos, la medida de similitud que se haya escogido con anterioridad y la puntuación de un usuario al producto similar.

$$pred(u, i_1) = \overline{rating_{i_1}} + \frac{\sum_{i_2} sim(i_1, i_2) * (rating_{i_2,u} - \overline{rating_{i_2}})}{\sum_{i_2} sim(i_1, i_2)} \quad (4.3)$$

$donde\ i_1\ es\ el\ primer\ producto, i_2\ es\ el\ segundo, u\ es\ el\ usuario$

Los N mejores resultados serán los que se recomendarán al usuario.

4.2.2 Modelos de factores latentes

Uno de los métodos con mejores resultados de los modelos de factores latentes es la factorización matricial (FM). La idea principal de este método consiste en factorizar una matriz de grandes dimensiones para construir otras dos de menores dimensiones, cuyo producto será la matriz original. Los modelos de FM unen los usuarios con los productos en un espacio de dimensionalidad reducido. Cada artículo está asociado a un vector q_i y cada usuario está asociado a un vector p_u. El producto de estos dos vectores es la interacción entre usuario y producto y será una puntuación, lo más próxima posible, de este usuario a ese producto.

$$\hat{r}_{ui} = q_i^T p_u. \quad (4.4)$$

$q_i \rightarrow representación\ del\ producto\ i$ $p_u \rightarrow representación\ del\ usuario\ u$

Estos dos vectores que se forman son los llamados **factores latentes** [1]. Mediante ellos podemos reducir la dimensionalidad de nuestra matriz inicial a una matriz final en la que solo aparezcan las preferencias de los usuarios.

En la siguiente ilustración podemos observar un ejemplo de un diagrama de factores latentes sacado del artículo *"Matrix Factorization Techniques for Recommender Systems"* [1].

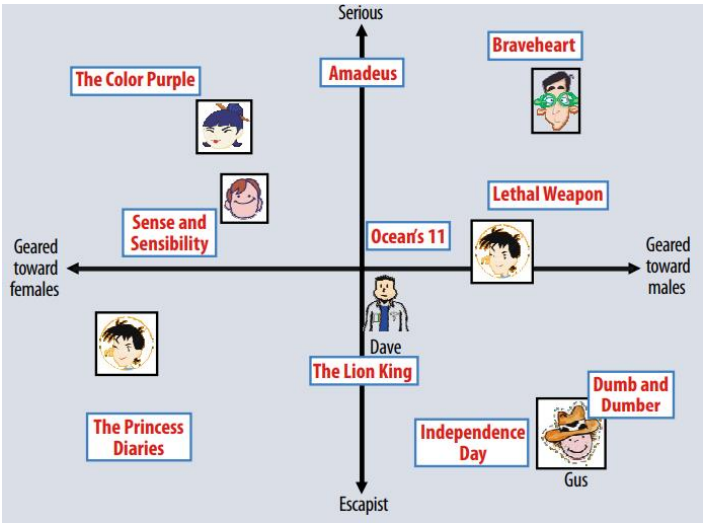

Figura 4.1. *Ejemplo factores latentes Netflix Price [1]*

El mayor esfuerzo de las técnicas de FM reside en calcular estos factores, es decir, obtener los vectores q_i y p_u. Para realizarlo se puede usar la ***Descomposición en valores singulares (SVD)*** [25]. SVD es una técnica de algebra matricial que permite descomponer una matriz en el producto de otras tres.

Esta técnica consiste en la descomposición de una matriz *A* con dimensiones *m x n* en tres matrices distintas *U [m x r], Σ [r x r], V [r x n]*. Donde U será la matriz que contenga los usuarios, V la matriz que contenga los productos y Σ será una matriz diagonal que soló tendrá elementos no nulos en su diagonal principal. Estos elementos representan la fuerza de cada factor latente. Si de la matriz Σ quitamos los elementos de menor valor, obtenemos la matriz original. De esta manera el algoritmo quedaría de la siguiente forma:

$$A = U\ \Sigma\ V^T \tag{4.5}$$

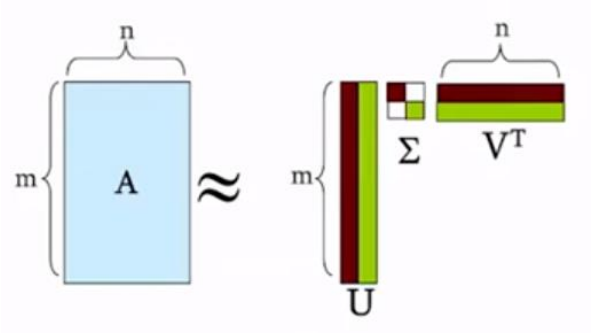

Figura 4.2. *Descomposición en valores singulares [25]*

Una vez obtenemos estas tres matrices podemos analizar los resultados para saber si verdaderamente estos factores latentes caracterizan de una manera adecuada a la matriz inicial. Se va a utilizar un ejemplo dentro del curso que la Universidad de Stanford imparte en *Coursera sobre Sistemas de Recomendación* [25] adaptándolo al caso de la lista de la compra.

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} = \begin{bmatrix} \mathbf{0.13} & 0.02 & -0.01 \\ \mathbf{0.41} & 0.07 & -0.03 \\ \mathbf{0.55} & 0.09 & -0.04 \\ \mathbf{0.68} & 0.11 & -0.05 \\ 0.15 & \mathbf{-0.59} & \mathbf{0.65} \\ 0.07 & \mathbf{-0.73} & \mathbf{-0.67} \\ 0.07 & \mathbf{-0.29} & \mathbf{0.32} \end{bmatrix} x \begin{bmatrix} \mathbf{12.4} & 0 & 0 \\ 0 & \mathbf{9.5} & 0 \\ 0 & 0 & \mathbf{1.3} \end{bmatrix} x \begin{bmatrix} \mathbf{0.56} & \mathbf{0.59} & \mathbf{0.56} & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & \mathbf{-0.69} & \mathbf{-0.69} \\ 0.40 & \mathbf{-0.80} & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

Figura 4.3*. Ejemplo factorización matricial [25]*

La matriz *A* corresponde a usuarios que han valorado una serie de productos, las filas de esta matriz serán los distintos usuarios mientras que las columnas serán los distintos productos. Podemos separar esta matriz en dos categorías de artículos para poder ver el ejemplo de los factores latentes. Las cuatro primeras filas corresponden a la categoría de Ultra Procesados mientras que las tres últimas corresponden a Productos Naturales.

Al analizar los resultados llegamos a las siguientes conclusiones. La primera columna de la matriz U, que corresponde a la categoría de ultra procesados, indica que los cuatro primeros usuarios tienen una relación bastante alta con esta categoría. Por otro lado, con la segunda columna podemos llegar a la conclusión de que corresponde a la categoría de

productos naturales puesto que los valores para los tres últimos usuarios son muy negativos. Como se puede apreciar en la matriz inicial, esos usuarios han calificado los productos ultra procesados con una puntuación muy baja, por ello la similitud de los artículos de esta categoría es negativa.

En cuanto a la matriz *V*, se puede observar algo similar a los usuarios. En las tres primeras columnas los valores son altamente positivos, ello indica que los primeros tres productos corresponden a la categoría de ultra procesados. En el caso contrario se encuentran las dos últimas columnas que indican que los dos últimos productos son productos naturales.

Por último, la matriz sigma que contiene los valores singulares, nos indica la fuerza de cada concepto dentro de la matriz original. Lo que se puede deducir es que cada valor en la diagonal principal corresponde a una categoría. Los ultra procesados serían el primer valor puesto que hay más usuarios que han puntuado mejor los productos pertenecientes a esa rama. El segundo valor podría ser el de productos naturales.

Como se puede observar en todas las matrices hay una fila (en U), columna (en V), o valor (en sigma) que no corresponde a ninguna categoría. Esto se puede interpretar como simple ruido que se introduce en la matriz final, ya que en la matriz sigma no tiene un valor significativo y en las otras dos matrices tiene valores con poco significado.

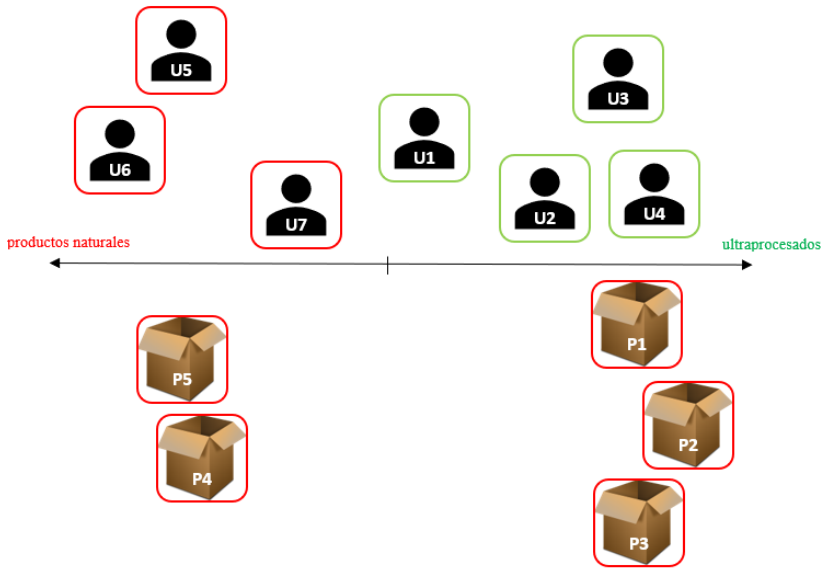

Figura 4.4. *Diagrama de factores latentes para el caso de estudio*

Esta es una técnica de algebra en la que no tienen cabida los sistemas de recomendación debido a la gran cantidad de posiciones nulas que contiene la matriz de cualquier SR. Cuando la matriz inicial está completa el procedimiento es el que se ha especificado anteriormente, pero en el caso contrario SVD no tiene manera de interpretar las posiciones vacías por tanto no se puede usar para el desarrollo de SR.

4.2.3 Algoritmos de aprendizaje

Ya que el SVD no es válido para problemas en los cuales se disponga de una matriz dispersa, existe un algoritmo que permite hacer una factorización de matrices en matrices dispersas. Este algoritmo se denomina ***mínimos cuadrados alternantes*** (ALS) [26]**.**

El ALS es un proceso iterativo de dos pasos. Si partimos de la ecuación 4.5 de SVD primero se fija la matriz *U* y se calcular la matriz *V*, una vez este proceso ha acabado se fija *V* y se calcula *U*. Este proceso se hace con cada fila y cada columna de ambas matrices.

$$\min_{x,y} \sum_{r_{u,i}\, conocido} \left(r_{u_i} - x_u^T y_i\right)^2 + \lambda(\|x_u\|^2 + \|y_i\|^2) \qquad (4.6)$$

Cada iteración para ajustar la matriz U o V es un proceso similar al de una regresión en el que se intentan ajustar los datos a una línea. La solución de cada regresión viene dada por el algoritmo de **Mínimos Cuadrados**, que garantiza que en cada paso la función de coste puede decrecer o permanecer estable pero nunca incrementarse. En la ecuación 4.6 que corresponde al ALS se puede observar como el primero de los términos $(r_{ui} - x_u^T y_i)^2$ es el del algoritmo de mínimos cuadrados mientras que el segundo $\lambda(\|x_u\|^2 + \|y_i\|^2)$ es un término de regularización para que no haya sobreajuste. El algoritmo de regresión con mínimos cuadrados se basa en aproximar los datos dados a una línea, medir la suma del cuadrado de las distancias de cada punto a la línea y obtener la mejor aproximación minimizando ese valor.

En **ALS** haremos lo mismo, pero alternando entre la matriz *U* y la matriz *V*. Para el caso de los usuarios la actualización de valores se hará según la ecuación 4.7 y para el caso de los productos se hará según la ecuación 4.8[26].

$$x_u = (Y^\top Y + \lambda I)^{-1} Y^\top r_u \tag{4.7}$$

$$y_u = (X^\top X + \lambda I)^{-1} X^\top r_i \tag{4.8}$$

Alternar entre los dos pasos garantiza la reducción en la función de coste hasta que converja. En la mayoría de las ocasiones cuando se trabaja con datos implícitos se establecen unas preferencias y niveles de confianza [27, 28]. Con las **preferencias** hacemos que los valores sean solo binarios. Si un usuario ha comprado un producto del supermercado se le asigna un uno que significa que le ha gustado. Por el contrario, si un usuario no ha comprado jamás un cierto producto se le asigna un cero para descartar ese valor.

$$p_{u,i} = \begin{cases} 1 & \text{if } r_{u,i} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{4.9}$$

Cuando asignamos un cero a las puntuaciones que son nulas le estamos dando un peso nulo dentro del SR puesto que al usuario no le ha gustado el producto y esto puede no ser del todo correcto. Puede que los valores nulos en una puntuación no signifiquen que al usuario no le ha gustado el producto, sino que no conoce el producto. Esto se soluciona estableciendo una nueva ecuación para los **niveles de confianza**. De esta manera, un artículo con puntuación nula tendrá un nivel de confianza mínimo de uno, mientras que el resto de los artículos puntuados tendrán un nivel de confianza que irá incrementándose. Este incremento está moderado por α, por tanto, α refleja cuanto valor se les da a los artículos que han sido comprados contra los artículos que no han sido comprados.

$$c_{u,i} = 1 + \alpha r_{u,i} \tag{4.10}$$

En la ecuación 4.10 el parámetro α se puede ajustar mediante los procesos de validación cruzada o con una validación simple. Una vez hecho este cambio la nueva función de coste que queremos minimizar es la que se muestra en la ecuación 4.11. En esta expresión se introducen el término de nivel de confianza $c_{u,i}$ para cada puntuación de cada usuario a cada producto y el término de preferencia $p_{u,i}$ para cada producto.

$$\min_{x,y} \sum_{r_{u,i} \text{is known}} c_{u,i}(p_{u,i} - x_u^\top y_i)^2 + \lambda(\sum_u ||x_u||^2 + \sum_i ||y_i||^2) \tag{4.11}$$

Se puede apreciar que existe una constante de regularización $\boldsymbol{\lambda}$ en todas las expresiones del ALS. El valor de esta constante se puede ajustar mediante la técnica de validación cruzada explicada en el apartado 3.4 para obtener mejores resultados del SR. Dado que el tiempo de cómputo del algoritmo crece de manera significativa al añadir este proceso, en algunas ocasiones no es necesario realizarlo. En el capítulo 5 se verá si ha sido necesario realizarlo en el caso concreto de nuestra BBDD.

4.3 Medidas para la precisión de las predicciones

La precisión de las predicciones dadas por el SR es uno de los aspectos más importantes ya que mide la calidad del SR.

4.3.1 Métricas para la ayuda en la toma de decisiones

Estas métricas [29] miden la capacidad de un SR de ayudar a un usuario a tomar buenas decisiones. Dentro de esta categoría se encuentran la **precisión** y el **recall**.
La precisión es el porcentaje de elementos dentro del total de seleccionados que son relevantes y el recall es el porcentaje de elementos relevantes que han sido seleccionados dentro del total de elementos relevantes.

$$precision = \frac{numero\ de\ elementos\ relevantes\ y\ recomendados}{numero\ de\ elementos\ recomendados} \tag{4.14}$$

$$recall = \frac{numero\ de\ elementos\ relevantes\ \ y\ recomendados}{numero\ de\ elementos\ relevantes} \tag{4.15}$$

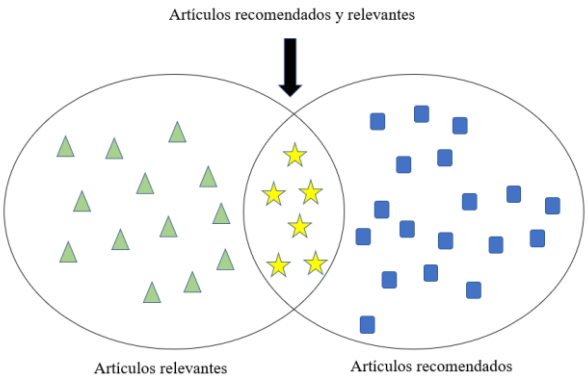

Figura 4.5. *Muestra gráfica de los conjuntos que se utilizan para calcular la precisión y el recall*

El objetivo de la medida de precisión es que seleccionemos solo los elementos que nos serán útiles para hacerle más fácil la decisión al usuario. La conclusión de esta medida es que hay más elementos relevantes en el SR de los que se quiere. Sin embargo, el objetivo de la medida de recall es no perder elementos que pueden ser útiles en el SR. Todas estas métricas se verán mejor con el siguiente ejemplo.

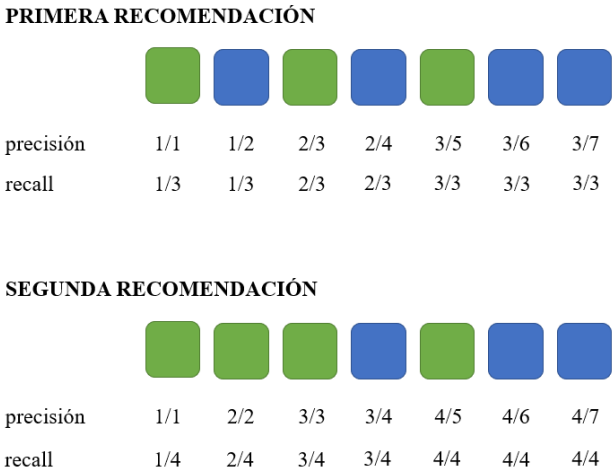

Figura 4.6. *Ejemplo para la explicación de las métricas de precisión y recall*

Existe una medida que considera ambas la precisión y el recall, se denomina **F1-score** y es la media armónica de estas dos medidas.

$$F1 - score = \frac{2 * precision * recall}{precision + recall} \tag{4.16}$$

4.3.2 Métricas para la predicción de la precisión en puntuaciones

En este apartado es necesario separar las métricas que se van a explicar en dos apartados. Esto es así puesto que las puntuaciones de un SR pueden provenir de información implícita o de información explícita. Para cada tipo existen unas métricas que se ajustarán más.

4.3.2.1 Puntuaciones explícitas

Para calcular la precisión de la predicción de las puntuaciones [30] existen tres métricas: **Error absoluto medio** (MAE), **Error cuadrático medio** (MSE) y **Raíz de la Desviación Cuadrática Media** (RMSE). El **MAE** mide la diferencia entre la puntuación predicha y la real, eliminar el signo introduciendo la operación de valor absoluto y se expresión es media puesto que se divide entre el número total de elementos en el conjunto.

$$MAE = \frac{\sum_{ratings} |\hat{r} - r|}{\#\, ratings} \tag{4.17}$$

El **MSE** eleva al cuadrado la diferencia entre la puntuación predicha y la real por dos razones: para poder eliminar el signo como en el MAE y para penalizar los errores grandes. Por ejemplo, si tenemos un error de 0.4 al elevarlo al cuadrado el resultado será 0.16, en cambio sí tenemos un error de 3 al elevarlo al cuadrado el resultado será 9.

$$MSE = \frac{\sum_{ratings} (\hat{r} - r)^2}{\#\, ratings} \tag{4.18}$$

El problema que tiene esta medida es que al estar elevada al cuadrado el resultado no va a estar dentro de la escala de puntuaciones reales. Por ese motivo se utiliza el **RMSE**. Esta medida penaliza los errores grandes y elimina el signo al igual que el MSE, pero al hacer la raíz cuadrada del resultado, este se encuentra dentro de la escala de puntuaciones.

$$RMSE = \sqrt{\frac{\sum_{ratings} (\hat{r} - r)^2}{\#\, ratings}} \tag{4.19}$$

4.3.2.2 Puntuaciones implícitas

Ya que nuestra BBDD consta de información implícita, estas métricas no serán las que se usarán. En el caso concreto de este proyecto se está interesado en evaluar una lista de elementos recomendados. Esto se conoce como **recomendaciones top-K.** Dos de las métricas más populares son el **mAP** y el **NDCG.**

El mAP medirá cuál es la eficiencia del SR. Para poder explicar el **mAP** [31,32] primero es necesario explicar la **precisión media** (AP). En el siguiente ejemplo [33] se calcula la precisión para cada elemento de una recomendación donde los elementos en verde son relevantes y los azules son no relevantes.

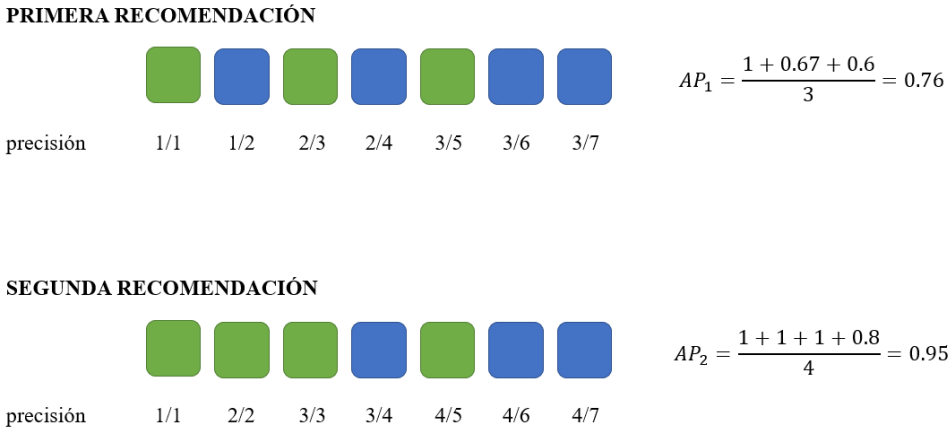

Figura 4.7. *Ejemplo para la explicación de la métrica de precisión media en dos recomendaciones distintas [33]*

Como se ha visto en el apartado anterior la precisión se calcula como el número de elementos recomendados y relevantes (los elementos verdes) entre el número total de elementos recomendados. Posteriormente, se calcula la precisión media recopilando la precisión de los elementos de color verde y dividiendo su suma entre el número total de elementos de color verde.

Una vez se tiene la precisión media de dos recomendaciones, se puede calcular el mAP como la media de AP para todos los usuarios.

$$mAP = \frac{\sum_{j=1}^{usuarios} (AP)_j}{\#\, usuarios} \tag{4.20}$$

La métrica denominada nDCG es la versión normalizada de Discounted Cumulative Gain (DCG) [33,34], por tanto, se comenzará explicando esta medida. Hasta ahora solo se han expuestos métricas en las que la importancia que se le da a cada resultado es la misma. En DCG cada artículo tiene una cierta importancia o un peso dentro de los resultados, ese

es el motivo por el que esta métrica contiene la palabra ganancia *(gain)*. La ganancia de los artículos que no han sido puntuados será nula.

$$DCG_k = \sum_{item=0}^{k} \frac{importancia_{item}}{\log_2(item+1)} \tag{4.21}$$

Lo que se quiere obtener es un ranking en el que los artículos que se posicionen en puestos altos tengan sean más relevantes que los que están en posiciones bajas. Para ello se utiliza una función como el logaritmo que tiene un incremento suave y provocará un descenso suave de la función del DCG. Con el ejemplo de la Figura 4.8 se explicará con mayor claridad. En este caso se tienen siete artículos con distintas puntuaciones.

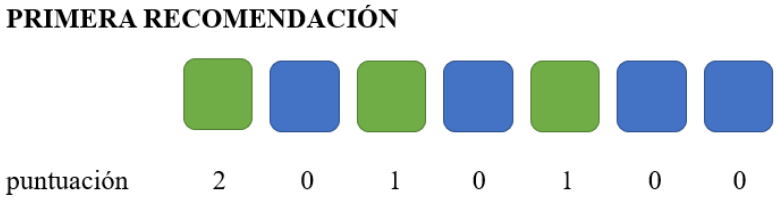

Figura 4.8. *Ejemplo para explicar la métrica DCG*

$$DCG_1 = \frac{2}{\log_2 1} + \frac{0}{\log_2 2} + \frac{1}{\log_2 3} + \frac{0}{\log_2 4} + \frac{1}{\log_2 5} + \frac{0}{\log_2 6} + \frac{0}{\log_2 7} = 3.06$$

A continuación, para calcular la versión normalizada (nDCG) se necesita el caso ideal en el que las puntuaciones estén ordenadas de mayor a menor.

$$DCG_{ideal_1} = \frac{2}{\log_2 1} + \frac{1}{\log_2 2} + \frac{1}{\log_2 3} + \frac{0}{\log_2 4} + \frac{0}{\log_2 5} + \frac{0}{\log_2 6} + \frac{0}{\log_2 7} = 3.63$$

Por tanto, la función de la versión normalizada de DCG es la que aparece en la siguiente expresión:

$$nDCG = \frac{DCG}{DCG_{ideal}} \tag{4.22}$$

Y para el caso ejemplo que se ha realizado, el resultado será:

$$nDCG_1 = \frac{3.06}{3.63} = 0.84 \tag{4.23}$$

En nDCG se medirá la similitud del SR con un caso ideal en el que todos los elementos de la lista de recomendación sean relevantes.

La principal diferencia [35] entre mAP y nDCG es que, como nDCG está modelado por un factor $\frac{1}{\log(r+1)}$ su pendiente decreciente es mucho más suave que la del mAP. En la siguiente figura podemos observarlo:

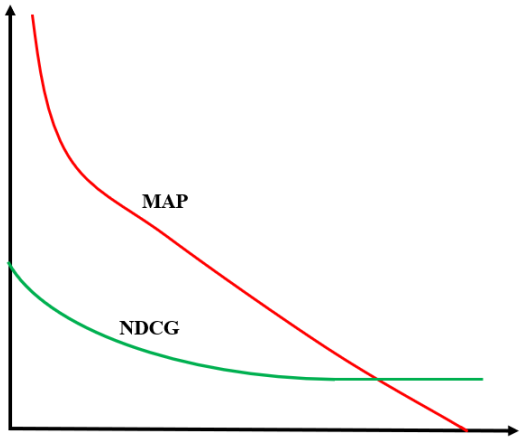

Figura 4.9. *Diferencia de pendiente entre el mAP y nDCG [35]*

5 IMPLEMENTACIÓN DEL SISTEMA DE RECOMENDACIÓN

En este capítulo se expone la implementación del sistema de recomendación completo. El capítulo está organizado de la siguiente manera. Primero, presento la implementación del sistema de recomendación con FC, haciendo hincapié en los algoritmos utilizados, los problemas y resultados de estos.

En la segunda parte se desarrolla la implementación del sistema de recomendación basado en contenido, explicando los algoritmos que se han utilizado.

5.1 Implementación del algoritmo ALS manualmente

No se ha utilizado ninguna librería para el ALS, sino que se ha implementado.

Como primer paso para comprobar el buen funcionamiento del algoritmo ALS se debe comprobar el progreso del error RMSE en cada iteración. Si este valor decrece con el paso de las iteraciones significará que el algoritmo funciona correctamente. A continuación, se puede apreciar la gráfica que demuestra que con cada iteración hay un descenso. En las cinco primeras iteraciones es cuando el error sufre un descenso considerable. A partir de la iteración número 15 la función de coste permanece más o menos estable puesto que se ha llegado a un mínimo local y esta función ya converge.

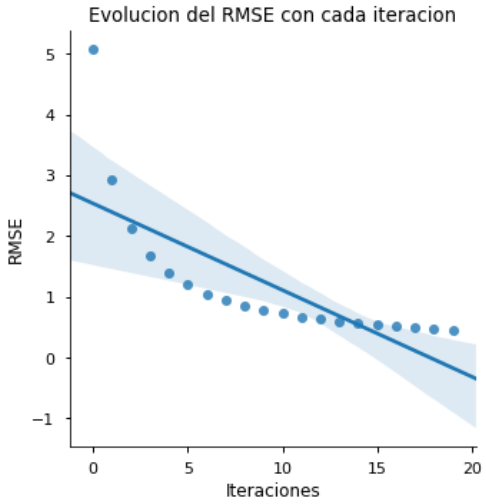

Figura 5.1. *Evolución del error cuadrático medio con el número de iteraciones*

Para asegurarnos de que el resultado que obtenemos sea el correcto y el algoritmo converja, el número de iteraciones que se selecciona es 25. En el siguiente apartado se aportará más información sobre esta elección.

5.1.1 Ajuste de parámetros

Como se ha explicado con anterioridad, dentro del ALS se introducen dos regularizaciones para así poder ajustar tanto la matriz de factores latentes de usuarios como la de productos. En estas regularizaciones existe un parámetro $\boldsymbol{\lambda}$ el cual hay que ajustar para poder obtener los mejores resultados.

La mejor técnica para ajustar este parámetro es validación cruzada en la cual se divide el conjunto total de datos en particiones. Estas particiones van cambiando y seleccionando conjuntos de datos distintos. Para poder realizar esta técnica es necesario bastante tiempo ya que se tendrían que calcular los resultados del algoritmo para cada subdivisión que se haga. Por ejemplo, si se elige dividir los datos en 10 particiones, habría que multiplicar el tiempo que tarda el algoritmo en ejecutarse y obtener los resultados por 10, ya que es el número de subconjuntos que existen. Una manera más sencilla de poder optimizar el valor de $\boldsymbol{\lambda}$ es mediante el valor de la métrica del mAP.

Las métricas que se van a utilizar en este apartado y el siguiente están basadas en los K primeros elementos de una lista de recomendación que se obtendrá. El primer paso para obtener resultados de las métricas es seleccionar la K que se va a usar. El valor de esta variable depende del caso de uso del SR. Como el proyecto se orienta hacia la recomendación de alimentos, y en las plataformas como Amazon o páginas web de comercio electrónico como las que se han expuesto en el marco socioeconómico de este proyecto se recomiendan entre 3 y 6 productos, se ha decidido seleccionar varios valores de K que serán 3,4,5,10. Se ejecutan varios procesos con distintos valores de $\boldsymbol{\lambda}$ y se comprueba el valor del mAP.

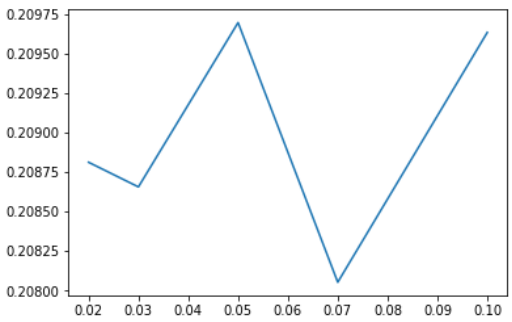

Figura 5.3.*Evolución de la métrica nDCG vs. parámetro de regularización (25 iteraciones)*

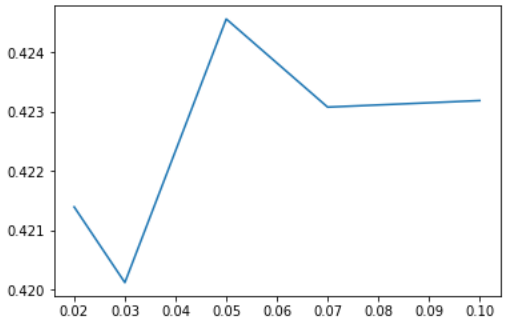

Figura 5.2*. Evolución de la métrica mAP vs. parámetro de regularización (25 iteraciones)*

En las Figuras 5.2 y 5.3 se observa la evolución del mAP y el nDCG con respecto a $\boldsymbol{\lambda}$**,** donde la "k" que se ha seleccionado es 3 puesto que seleccionando un valor más alto ya se incluyen las tres primeras recomendaciones. Como se puede ver en las figuras el valor del mAP más alto se alcanza cuando $\boldsymbol{\lambda} = 0.05$ ó $\boldsymbol{\lambda} = 0.1$ por lo que habría que escoger entre estos dos valores. Pero al observar la figura 5.3 se toma la decisión de elegir $\boldsymbol{\lambda} = 0.05$ puesto que el valor del nDCG es mayor.

Por tanto, se utilizará un parámetro de regularización $\boldsymbol{\lambda}$ **= 0.05.** Los resultados son lógicos puesto que las puntuaciones que se insertan en la matriz tienen valores pequeños y si se escogiesen valores muy altos de regularización con respecto al de las puntuaciones, quedarían reducidos la mayoría de los valores de la matriz final.

Otro punto importante en el desarrollo del algoritmo es la selección de factores latentes. El valor que se quiera fijar de factores latentes influye en el tiempo de cómputo del algoritmo ya que cuanto mayor número de factores latentes se fijen, mayor dimensión tendrá la matriz de usuarios y la de productos. Para la obtención del número de factores óptimo para este SR se realizan la siguientes gráficas en las que se muestra el cambio que sufren las métricas mAP y nDCG respecto a los factores latentes.

En el primer apartado de este capítulo se ha plasmado el número de iteraciones que será conveniente seleccionar para que el algoritmo converja, 25 iteraciones. En las siguientes gráficas se han realizado experimentos con 20, 25 y 30 iteraciones para poder demostrar gráficamente que los resultados para 20 y 30 iteraciones no son los esperados.

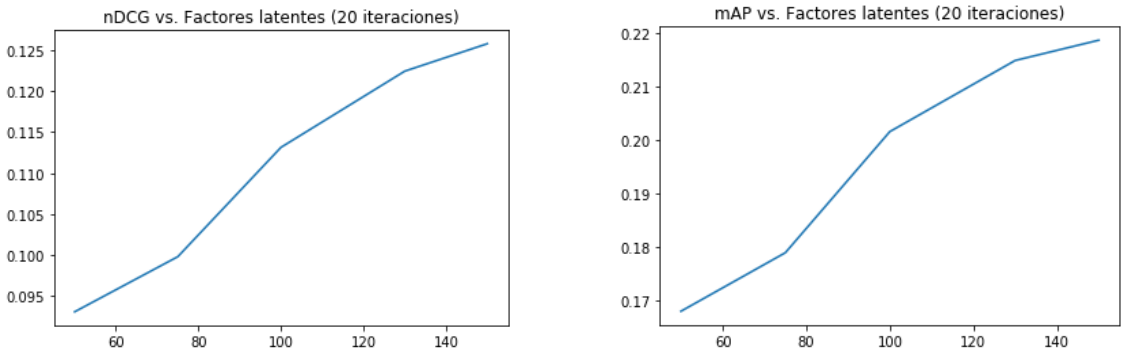

Figura 5.4. *Evolución de las métricas nDCG y mAP con respecto a los distintos valores de factores latentes para 20 iteraciones del algoritmo*

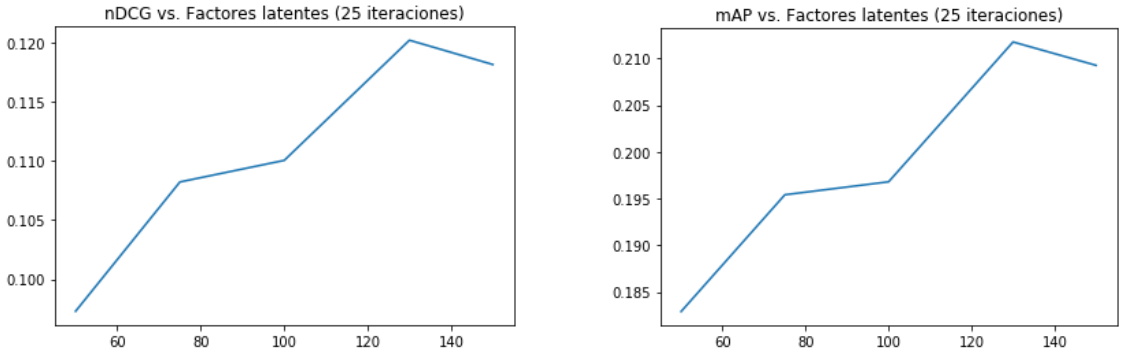

Figura 5.5. *Evolución de las métricas nDCG y mAP con respecto a los distintos valores de factores latentes para 25 iteraciones del algoritmo*

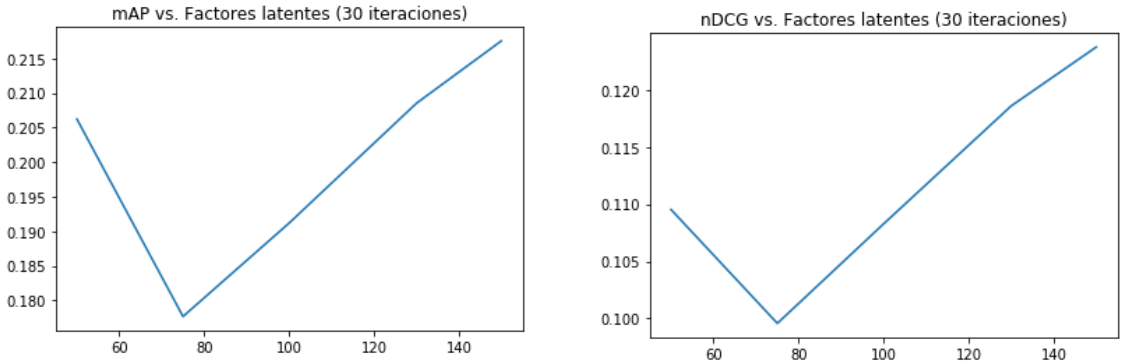

Figura 5.6. *Evolución de las métricas nDCG y mAP con respecto a los distintos valores de factores latentes para 30 iteraciones del algoritmo*

En la Figura 5.4 se muestra un comportamiento del mAP y el nDCG que podría darse por válido ya que para valores pequeños de factores latentes el SR no tiene suficientes datos y por tanto los valores de ambas métricas son pequeños. Para el caso de que el número de factores latentes sea elevado (150 factores latentes) se podría interpretar que no se ha llegado al límite de factores latentes que admite el sistema para el cual los resultados empezarían a empeorar porque hay excesivos datos. Pero esta deducción varía al observar la Figura 5.5 puesto que se aprecia como para 150 factores latentes el mAP y el nDCG comienzan a decrecer. En cuanto a la Figura 5.6 en la que se muestra el rendimiento con 30 iteraciones del algoritmo, se demuestra que los resultados no tienen sentido puesto que para 50 factores latentes el mAP y el nDCG son mayores que para 75 factores.

De entre los tres pares de gráficas que se han mostrado, el comportamiento más lógico es de la Figura 5.5, con 25 iteraciones. Estas gráficas muestran que con un número pequeño de factores latentes (entre 50 y 75) el SR no tiene suficiente información sobre los usuarios y los productos al carecer del número de factores latentes necesarios. A medida que el número de factores latentes aumenta, el mAP también aumenta y por tanto se obtiene un mejor rendimiento del sistema. El valor máximo se alcanza en los 130 factores latentes y en el siguiente valor (150 factores latentes) el mAP comienza a descender. Este descenso se debe a que el SR tiene demasiada información y no se obtienen buenos resultados. Por todo ello, se seleccionará el valor de 130 factores latentes.

A continuación, se realizan una serie de experimentos sobre el conjunto de validación. En estos experimentos se compara el desarrollo del SR del proyecto con un SR aleatorio.

Modelo	mAP	nDCG	Precision	Recall	F1-score
k= 3 con SR del proyecto	0.208	0.421	0.0252	0.0010	0.0019
k=3 aleatorio	0.029	0.016	0.0146	0.0005	0.0011

Tabla 5.1. *RESULTADOS DE LAS MÉTRICAS MAP Y NDCG EN UNA LISTA DE RECOMENDACIÓN TOP-3 SOBRE EL CONJUNTO DE VALIDACION*

Modelo	mAP	nDCG	Precision	Recall	F1-score
k= 4 con SR del proyecto	0.228	0.439	0.0243	0.0013	0.0024
k=4 aleatorio	0.035	0.017	0.0151	0.0008	0.0015

Tabla 5.2*.RESULTADOS DE LAS MÉTRICAS MAP Y NDCG EN UNA LISTA DE RECOMENDACIÓN TOP-4 SOBRE EL CONJUNTO DE VALIDACION*

Modelo	mAP	nDCG	Precision	Recall	F1-score
k= 5 con SR del proyecto	0.243	0.458	0.0237	0.0016	0.0029
k=5 aleatorio	0.042	0.019	0.0148	0.0001	0.0018

Tabla 5.3. *RESULTADOS DE LAS MÉTRICAS MAP Y NDCG EN UNA LISTA DE RECOMENDACIÓN TOP-5 SOBRE EL CONJUNTO DE VALIDACION*

Modelo	mAP	nDCG	Precision	Recall	F1-score
k= 10 con SR del proyecto	0.262	0.536	0.0215	0.003	0.0050
k=10 aleatorio	0.047	0.017	0.0154	0.002	0.0036

Tabla 5.4. *RESULTADOS DE LAS MÉTRICAS MAP Y NDCG EN UNA LISTA DE RECOMENDACIÓN TOP-10 SOBRE EL CONJUNTO DE VALIDACION*

Como se aprecia en los resultados expuestos en las anteriores tablas el rendimiento del SR implementado en el proyecto es considerablemente superior al de un SR aleatorio, con un rendimiento casi diez veces más alto que el aleatorio.

5.1.2 Resultados del sistema de recomendación

Una vez se han ajustado los parámetros necesarios para el funcionamiento del algoritmo y se han obtenido resultados óptimos en el conjunto de validación, se procede a examinar el rendimiento real del SR del proyecto con el conjunto de test.

Como se ha explicado anteriormente se utilizarán 130 factores latentes, 0.05 como valor de regularización y 25 iteraciones.

	mAP	nDCG	Precisión	Recall	F1-score
k=3 en el modelo del proyecto	0.0517	0.0962	0.0188	0.005	0.0079

Tabla 5.5. *RESULTADOS DEL SISTEMA DE RECOMENDACIÓN EN EL CONJUNTO DE TEST CON LOS 3 PRIMEROS ELEMENTOS DE UNA LISTA DE RECOMENDACIÓN*

	mAP	nDCG	Precisión	Recall	F1-score
k=4 en el modelo del proyecto	0.0583	0.0988	0.0186	0.008	0.0112

Tabla 5.6*.RESULTADOS DEL SISTEMA DE RECOMENDACIÓN EN EL CONJUNTO DE TEST CON LOS 4 PRIMEROS ELEMENTOS DE UNA LISTA DE RECOMENDACIÓN*

	mAP	nDCG	Precisión	Recall	F1-score
k=5 en el modelo del proyecto	0.0632	0.1017	0.019	0.009	0.0122

Tabla 5.7*.RESULTADOS DEL SISTEMA DE RECOMENDACIÓN EN EL CONJUNTO DE TEST CON LOS 5 PRIMEROS ELEMENTOS DE UNA LISTA DE RECOMENDACIÓN*

	mAP	nDCG	Precisión	Recall	F1-score
k=10 en el modelo del proyecto	0.0773	0.1152	0.0193	0.019	0.0345

Tabla 5.8. *RESULTADOS DEL SISTEMA DE RECOMENDACIÓN EN EL CONJUNTO DE TEST CON LOS 10 PRIMEROS ELEMENTOS DE UNA LISTA DE RECOMENDACIÓN*

Haciendo hincapié en los resultados de la medida nDCG para 10 elementos en la lista de recomendación (Tabla 5.8), se puede deducir que el modelo está sobre ajustado puesto que los resultados de validación para este mismo caso son 4 veces mejor que los del sistema real. En el caso ideal, los resultados de test deberían ser muy parecidos a los obtenidos en validación. Esto es algo que era probable puesto que solo se ha procedido al ajuste de parámetros por validación. En caso de querer mejorar las prestaciones del sistema real (en test), habría que desarrollar un proceso de validación cruzada. Este proceso no se ha implementado en el proyecto debido a la cantidad de tiempo de cómputo que se necesita.

La lista de recomendación es mejor a medida que se van añadiendo más elementos puesto que es más fácil encontrar elementos relevantes entre más elementos recomendados. Tanto los valores de la precisión como el de recall no varían significativamente en los conjuntos de test y validación, esto significa que el SR real encuentra el mismo número de elementos relevantes dentro de una recomendación que los que encuentra en los experimentos. Con estos resultados llegamos a la conclusión de que la lista con los 10 mejores elementos se asemeja un 11 % a una lista ideal, lo cual es un resultado que se podría mejorar introduciendo una serie de mejoras.

5.1.3 Resultados del sistema real para los dos usuarios con mayores puntuaciones

Como se ha explicado en el Capítulo 3.7, al analizar los datos se ha descubierto que hay dos usuarios que asignan a casi todos los productos que adquieren la máxima puntuación, es decir, compran diariamente cada producto. Para estos usuarios el rendimiento del sistema debería ser muy superior al del resto de usuarios puesto que todos los productos que se encuentran en su lista de la compra se pueden entender como relevantes ya que se adquieren con mucha frecuencia.

	mAP@3	**nDCG@3**
Modelo para usuarios más activos	0.6933	0.75

Tabla 5.9. *Resultados del rendimiento de una lista de recomendación para los usuarios más activos*

Al obtener los resultados podemos afirmar que, en el SR propuesto en el proyecto, en el caso de estos dos usuarios hay un 75% de similitud entre sus listas de recomendación y una lista de recomendación ideal en la que todos los productos sean relevantes. Se ha de hacer hincapié en que no se alcanza la perfección puesto que algunos de los productos comprados por estos usuarios no están calificados con la puntuación más alta.

5.1.4 Recomendaciones a usuarios y productos similares

La implementación del algoritmo ALS mediante la librería "Implicit" también contiene una recomendación de productos similares. Para esta recomendación se usará una función programada por la propia librería con los resultados extraídos del ALS implementado manualmente. La recomendación de productos similares, como se ha explicado en el Capítulo 5 se hará mediante el nombre de cada producto o la colocación en el supermercado de los productos. Un ejemplo de esta recomendación se muestra a continuación. En el ejemplo se elige el producto con identificador 300 que corresponde a "Organic Enriched Unbleached White Flour", que se sitúa en el pasillo 17 y el departamento 13.

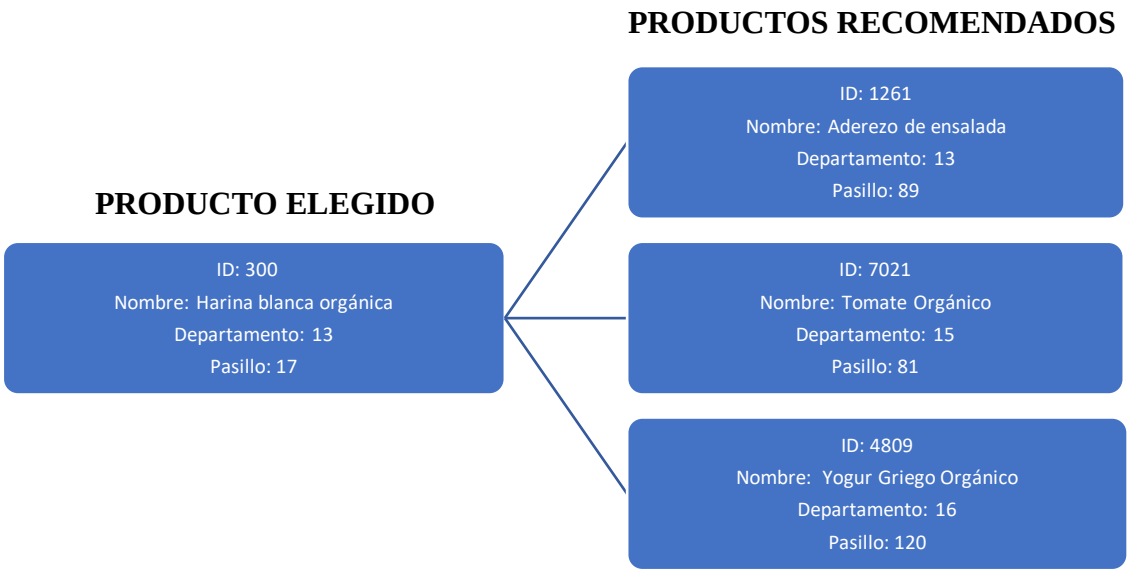

Figura 5.7. *Representación de las recomendaciones que se obtienen del ALS*

En la recomendación se observa como el primer resultado es más relevante puesto que se encuentra en el mismo departamento que el producto elegido. Los otros dos resultados tienen menos relevancia ya que solo comparten una palabra en el nombre del producto y se encuentran en departamentos cercanos al producto elegido.

6 GESTIÓN DEL PROYECTO

En este capítulo se muestra la planificación del proyecto, enumerando las tareas que han sido necesarias para la realización de este. También se añade un presupuesto total orientativo.

6.1 Planificación del proyecto

La realización del proyecto ha consistido en las siguientes fases:

- BASE DE DATOS. Búsqueda de una base de datos en la cual poder introducir un sistema de recomendación.
- INVESTIGACION. Estudio sobre los SR y los algoritmos que se utilizan para su implementación.
- ANALISIS DE LA BASE DE DATOS. Analizar la base de datos encontrada para poder introducirla de manera óptima en el sistema de recomendación.
- DESARROLLO DE LOS ALGORITMOS. Programación en Python de los distintos algoritmos.
- DOCUMENTACIÓN. Por último, realizo la documentación del proyecto para dejar constancia del progreso.

6.2 Análisis de costes

En este apartado se detallan los costes totales del proyecto. Con la intención de que la información quede expuesta de forma clara, los costes se dividirán en varios apartados: coste del personal y coste de mantenimiento.

6.2.1 Personal

Para la realización de este proyecto y su posterior puesta en el mercado se necesitan:

- Un analista que será el encargado de investigar, diseñar y desarrollar el proyecto. Su desempeño en este proyecto consistirá en una jornada laboral de 8 horas diarias. El sueldo del analista será de 15 euros por hora.

- Un consultor senior que será el encargado de poner en el mercado el proyecto para que otras empresas presenten ofertas. También tendrá la función de supervisar el trabajo del analista. Su trabajo consistirá en una jornada laboral de 8 horas diarias y su sueldo será de 30 euros por hora.

Tarea	Analista	Senior
Definición de los objetivos del proyecto		10 h
Investigación		
Tipos de sistemas de recomendación	22 h	
Algoritmos para los sistemas de recomendación	70 h	
Análisis de la base de datos		
Auditoría de datos	32 h	
Diseño del algoritmo		
Definición de las etapas de desarrollo del algoritmo	6 h	6 h
Análisis del algoritmo	30 h	
Implementación del algoritmo		
Desarrollo del algoritmo	60 h	
Entrenamiento	12 h	
Pruebas	3 h	
Integración		
Resultados	5 h	2 h
Mantenimiento	1 h	
Documentación del proyecto	80 h	
TOTAL	64 días	6 días

Tabla 6.1*.Tareas del personal*

	Analista	Senior
Días totales	64 días	6 días
Horas / día	5	3
Horas totales	320 horas	18 horas
€ / hora	15 € / hora	30 € / hora
Total, del personal	4.800 €	540 €
TOTAL	**5340 €**	

Tabla 6.2*.Coste del personal*

6.2.2 Presupuesto total

Dentro de los gastos de material se incluye un portátil marca HP para cada miembro del equipo valorado en 1.300 €, así como pantallas para mejorar su trabajo valoradas en 200€ y demás material necesario para cualquier oficina. El proyecto se desarrollará desde un espacio alquilado dentro de las oficinas de la empresa de co-working WeWork en el Paseo de la Castellana. El coste del alquiler de un espacio de trabajo para 3 ó 4 personas es de 1700 €/mes. Como el desarrollo total del proyecto nos llevará en torno a 3 meses el coste total será de 5100€.

Coste material	5.000 €
Coste personal	5.340 €
Coste	5.100 €
TOTAL	**15.440 €**

Tabla 6.3*.Coste total del proyecto*

7 CONCLUSIONES

Este capítulo consta de dos puntos. En el primero de ellos se expone la conclusión al proyecto. Finalmente, el último punto del trabajo se basa en las posibles mejoras que se podrían incorporar.

7.1 Conclusión

En este proyecto se ha desarrollado un SR con filtrado colaborativo para la lista de la compra con el objetivo de ayudar a los consumidores a que las decisiones a la hora de adquirir productos en un supermercado sean más fáciles de tomar. El sistema recomendará artículos que han sido calificados con una puntuación alta a usuarios similares a nuestro usuario activo.

La base de datos que se ha usado para desarrollar el FC proporciona información implícita sobre cada pedido de cada usuario. Así, mediante la frecuencia con la que se ha comprado cada producto se pueden averiguar los gustos de cada usuario. La información implícita no es del fiable puesto que se necesitarían muchos más datos sobre el comportamiento del usuario para poder elaborar un perfil más completo.

Una vez analizados los resultados en el Capítulo 5, se puede considerar que se han obtenido los resultados que se proponían al comienzo del proyecto. El SR, en los experimentos de validación, tiene mejores prestaciones que un sistema aleatorio de recomendación y dentro de una lista de recomendación de 3 elementos, el 40% de los elementos serán relevantes, es decir, de cada tres productos por lo menos uno será relevante. Otro de los objetivos a cumplir era la adaptación de los datos al algoritmo por medio de un amplio análisis de la base de datos. Este objetivo no está plenamente conseguido puesto que, según se puede observar en el Capítulo 5.1.2 con los resultados del SR real, la diferencia entre las métricas en el conjunto de validación y el de test son considerables, alcanzándose el 50 % de diferencia, por tanto, existe sobre ajuste.

La conclusión más importante es que, mediante un algoritmo diseñado manualmente (ALS), se han conseguido unos resultados aceptables.

7.2 Líneas futuras de trabajo

Durante la realización del proyecto han surgido posibles nuevas implementaciones con el fin de mejorar el sistema de recomendación para obtener mejores resultados, utilizar un mayor número de nuevas tecnologías, aprovechamiento de recursos en la nube…Algunas de las mejoras serían:

- Utilizar una máquina con una instancia de (Amazon Web Services) AWS para poder tener el SR por completo en la nube.
- Implementar el algoritmo con la tecnología de Spark para poder mejorar su rendimiento y rapidez de cómputo.
- Implementación de un SR basado en contenido que recomiende productos similares por pasillos o departamentos y así poder darles una utilidad a los artículos menos puntuados.
- Usar una base de datos que se actualice para poder hacer un SR en tiempo real.
- Conseguir una base de datos con más información acerca de los productos para poder ofrecer recomendaciones que tengan en cuenta propiedades como el precio o la marca.
- Ajustar el parámetro de regularización de la regresión Ridge mediante validación cruzada para poder tener un sistema que se ajuste mejor a cualquier otra base de datos.

BIBLIOGRAFÍA

[1] Y.Koren, R.Bell, C.Volinsky, "Matrix Factorization Techniques for Recommender Systems", *Data Jobs,* Agosto 2009. **[En línea]. Disponible en:** https://datajobs.com/data-science-repo/Recommender-Systems-[Netflix].pdf

[2] I.MacKenzie, C.Meyer, S.Noble, "How retailers can keep up with customers", *McKinsey&Company,* Octubre 2013. **[En línea]. Disponible en:** https://www.mckinsey.com/industries/retail/our-insights/how-retailers-can-keep-up-with-consumers

[3] A.Ibañez, "Avances en tecnología de recomendaciones para tiendas de comercio electrónico", *RTVE,* 6 septiembre 2014. **[En línea]. Disponible en:** http://www.rtve.es/noticias/20140906/avances-tecnologia-recomendaciones-para-tiendas-comercio-electronico/1006680.shtml

[4] A.Zhang, N.Fawaz, S.Ioannidis y A.Montanari, "Guess Who Rated This Movie: Identifying Users Through Subspace Clustering",*Cornell Univeristy Library,* 9 agosto 2014. **[En línea]. Disponible en**: https://arxiv.org/abs/1408.2055

[5] L.Alonso, "Amazon lanza su servicio Amazon Fresh por primera vez en Europa", *Marketing 4 ecommerce,* 14 junio 2016. **[En línea]. Disponible en:** https://marketing4ecommerce.net/amazon-lanza-servicio-amazon-fresh-primera-vez-europa/

[6] A. Nuñez-Torrón Stock, "El ecommerce de alimentación se multiplica por tres en España", *Ticbeat,* 20 junio 2018. **[En línea]. Disponible en:** http://www.ticbeat.com/tecnologias/ecommerce-supermercados-online-espana/

[7] G.Moreno, "Los supermercados consiguen que el ecommerce de alimentos se extienda en España", *Statista,* 19 junio 2018. **[En línea]. Disponible en:** https://es.statista.com/grafico/14334/los-supermercados-consiguen-que-el-ecommerce-de-alimentos-se-extienda-en-espana/

[8] Sin autor, "4 retos para el eCommerce de alimentación en España", *Marking 4 ecommerce,* 10 febrero 2017. **[En línea]. Disponible en:** https://marketing4ecommerce.net/4-retos-para-el-ecommerce-de-alimentacion-en-espana/

[9] D.Jannach, M.Zanker, A.Felfernig y G.Friedrich, "Collaborative recommendation" en *Recommender Systems: An introduction.* New York: Cambridge, 2011, 13 – 50.

[10] T. Hopmans, "A recommendation system for blogs: Setting up the prerequisites (part 1)", *The Marketing Techonologies,* 19 noviembre 2015. **[En línea]. Disponible en:** https://www.themarketingtechnologist.co/building-a-recommendation-engine-for-geek-setting-up-the-prerequisites-13/

[11] C.Pinela, "Recommender Systems – User-Based and Item-Based Collaborative Filtering", *Medium,* 6 noviembre 2017. **[En línea]. Disponible en:** https://medium.com/@cfpinela/recommender-systems-user-based-and-item-based-collaborative-filtering-5d5f375a127f

[12] F.Ricci, L.Rokach, B.Shapira, "Content-based Recommender Systems: State of the Art and Trends" en *Recommender Systems Handbook*. New York: Springer,2010, 51 – 79.

[13] D.Jannach, M.Zanker, A.Felfernig,G.Friedrich, "Content-based recommendation" en *Recommender Systems: An introduction.* New York: Cambridge, 2011, 13 – 50.

[14] S.Das, "Beginners Guide to learn about Content Based Recommender Engines", *Analytics Vidhya,* 11 agosto 2015. **[En linea]. Disponible en:** https://www.analyticsvidhya.com/blog/2015/08/beginners-guide-learn-content-based-recommender-systems/

[15] D.Jannach, M.Zanker, A.Felfernig,G.Friedrich, "Knowledge-based Recommendation" en *Recommender Systems: An introduction.* New York: Cambridge, 2011, 81 – 122.

[16] David W. Aha, "The omnipresence of Case-Based Reasoning in Science and Application", *Naval Research Laboratory,* noviembre 1998. **[En línea]. Disponible en:** http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.126.1155&rep=rep1&type=pdf

[17] R.P. Burgues, "Técnicas de Aprendizaje automático y Azure Machine Learning", *Pensando bajo la lluvia,* 2 agosto 2018. **[En línea]. Disponible en:**

https://rodrigopb.wordpress.com/2015/03/13/tecnicas-de-aprendizaje-automatico-y-azure-machine-learning/

[18] D.Jannach, M.Zanker, A.Felfernig,G.Friedrich, "Hybrid recommendation approaches" en *Recommender Systems: An introduction.* New York: Cambridge, 2011, 124 – 142.

[19] R. Burke, "Hybrid Recommender Systems: Survey and Experiments", California State University, Fullerton.**[En línea]. Disponible en:** http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.88.8200&rep=rep1&type=pdf

[20] Food Watch, "El 70 % de los productos alimenticios de un supermercado son ultraprocesados", *Food Watch,* 30 agosto 2017. **[En línea]. Disponible en:** https://www.foodwatch.org/nl/pers/persberichten/persberichten-detail/70-supermarkt-bestaat-uit-omstreden-ultra-processed-foods/

[21] J.I.Bagnato, "Qué es overfitting y underfitting y cómo solucionarlo", *Aprende Machine Learning,* 12 diciembre 2017. **[En línea]. Disponible en:** http://www.aprendemachinelearning.com/que-es-overfitting-y-underfitting-y-como-solucionarlo/

[22] J.Brownlee, "What is the Difference Between Test and Validation Datasets?", *Machine Learning Mastery,* 14 julio 2017. **[En linea]. Disponible en:** https://machinelearningmastery.com/difference-test-validation-datasets/

[23] [Cross validation] Udacity, "K-Fold Cross Validation – Intro to Machine Learning", *Youtube,* 23 febrero 2015. **[Video en línea]. Disponible en:** https://www.youtube.com/watch?v=TIgfjmp-4BA

[24] B.Sarwar, G.Karypis, J.Konstan y J.Riedl ,"Item-Based Collaborative Filtering Recommendation Algorithms", *Department of Computer Science and Engineering, University of Minnesota*. Minneapolis, mayo 2001. **[En linea]. Disponible en:** http://delivery.acm.org/10.1145/380000/372071/p285-sarwar.pdf?ip=193.147.77.72&id=372071&acc=ACTIVE%20SERVICE&key=DD1EC5BCF38B3699%2E45FCEA5C82CAB4B5%2E4D4702B0C3E38B35%2E4D4702B0C3E38B35&__acm__=1537356171_0980ba303ae12d969dd675f2fb881e65

[25] Stanford University. “Lecture 55 – Latent Factor Recommender System”. *Youtube,* 13 abril 2016. **[Video en línea]** https://www.youtube.com/watch?v=E8aMcwmqsTg

[26] Y. Hu, Y.Koren, C.Volinsky, “Collaborative Filtering for Implicit Feedback Datasets”, *2008 Eight IEEE International Conference on Data Mining,*Pisa, 2008, pp.263-272. **[En línea]. Disponible en:** http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4781121&isnumber=4781078

[27] D.Lee, “Collaborative Filtering using Alternating Least Squares”, *Daniel Nee,* 17 septiembre 2016. **[En línea]. Disponible en:** http://danielnee.com/2016/09/collaborative-filtering-using-alternating-least-squares/

[28] V. Köhler. “ALS Implicit Collaborative Filtering”. *Medium,* 23 agosto 2017. **[En línea]. Disponible en:** https://medium.com/radon-dev/als-implicit-collaborative-filtering-5ed653ba39fe

[29] U. de Minnesota. “Decision Support Metrics”. *Coursera.* **[Video en línea]. Disponible en:** https://www.coursera.org/learn/recommender-metrics/lecture/nnobM/decision-support-metrics

[30] U. de Minnesota. “Precision Accuracy Metrics”. *Coursera*. **[Video en línea]. Disponible en:** https://www.coursera.org/learn/recommender-metrics/lecture/LGLb3/prediction-accuracy-metrics

[31] S.Sawtelle, “Mean Average Precision (MAP) For Recommender Systems”, *SonyaSawtelle,* 25 octubre 2016. **[En linea]. Disponible en:** http://sdsawtelle.github.io/blog/output/mean-average-precision-MAP-for-recommender-systems.html

[32] V.Lavrenko, “Evaluation 12: mean average precision”, *Youtube,* 20 enero 2014. **[Video en línea]. Disponible en:** https://www.youtube.com/watch?v=pM6DJ0ZZee0

[33] D.Parra, “Métricas de Evaluación”, *DParra*. **[En línea]. Disponible en:** http://dparra.sitios.ing.uc.cl/classes/recsys-2017-2/clase5_metricas.pdf

[34] Zygmunt Z, “Evaluating recommender systems”, *FastML,*31 agosto 2015. **[En linea]. Disponible en:** http://fastml.com/evaluating-recommender-systems/

[35] V. Lavrenko, “Evaluation 13: MAP vs NDCG”, *Youtube*, 20 enero 2014. **[Video en línea]. Disponible en:** https://www.youtube.com/watch?v=qm1In7NH8WE

ANEXO I. ACRÓNIMOS

FC Filtrado colaborativo

CF Colaborative filtering

BC Basados en contenido

CB Content-based

KB Knowledge-based

FM Factorización matricial

MF Matrix Factorization

ALS Alternating Least Squares

SR Sistema de Recomendación

RS Recommender System

MSE Mean Squared Error / Error cuadrático medio

RMSE Root Mean Squared Error / Raíz de la Desviación Cuadrática Media

MAE Mean Absolute Error / Error absoluto medio

AWS Amazon Web Services

TF-IDF Term frequency – Inverse Document Frequency

CE Comercio electrónico

mAP Mean Average Precision

AP Average Precision

DCG Discount Cumulative Gain

nDCG Normalized Discount Cumulative Gain

8 EXTENDED SUMMARY

INTRODUCTION, MOTIVATION AND GOALS OF THE PROJECT

RS have been in our lives since ever because we make decisions lots of times in our routine. The first RS tried to suggest similar products based on what similar users liked and that is named collaborative filtering.

When ecommerce began to rise users had more issues when deciding because there was a huge quantity of items. The difference was minimum between two different products: the brand, the price or the quality. Based on this inconvenient a new category of RS was born, based in content. This type of RS restricted the number of options for each user and it was easier for them to choose one item.

In this project a RS is going to be implemented with the main goal of making users decision easier when doing the shopping list. Before the implementation, we have to study the theory of RS and the algorithms that can be used in each case. It has been done a previous analysis of the database to know more about the data and that way we can use it in the best way and obtain the best results.

The motivation of the project is to succeed in the product recommendations of a shopping list. If we can achieve this, it is going to be an improvement for enterprises and for customers. On one hand, the enterprises will sell products that, maybe in other cases were going to be forgotten or not seen. This fact is a good point for this companies because it is going to make benefits. Also, customers will be satisfied because they are going to buy what they wanted, and this is going to make loyal clients.

Food ecommerce is very developed in countries as United Kingdom or United States but here in Spain, the industry is old-fashioned. Since 2015 ecommerce has evolved in Spain and many users use it but, after doing a socio-economic analysis of the sector we see a lack of RS technology in these online selling platforms. There are only a few companies that have RS in their selling platforms and we want to change to modernize the sector.

STATE OF ART: RECOMMENDER SYSTEMS

There are four different categories of RS, and they will be explained in this chapter with all the techniques and methods used in each of them.

To begin with, the most important category in this project is **collaborative filtering** because the practical case of the project is based in this method. CF (Collaborative Filtering) principal idea is that users that are similar will also have the same preferences and the system will recommend the same products. An important point in this type of RS are the ratings because CF uses these ratings to know if a product is likely to be recommended to a user. Inside CF there are two approximations: user-based and item-based. User-based says that it is likely to happen that if two users agreed in the past about a product then they will also agree in the present. To predict the rating of this new product to recommend the system will use similarity measures such as Pearson correlation. The main issue of these systems is the **cold-start problem**. On the other hand, item-based says that users will have similar opinions of similar products. This system is comparable to content-based with the divergence that item-based is based on user's ratings. This approximation avoids the cold-start problem and improves scalability because similarity between items is more stable between users.

Content-based (CB) RS recommend items to users that have liked similar items in the past. To have data about these similar items there must be an item representation will all its features, technically speaking, we need descriptions and characteristics about every item to make these recommendations. Some advantages of CB are:

- *Users independence*
- *Transparency*
- *New products*

Oppositely, the drawbacks of these category of RS are:

- *Limited content analysis*
- *Over specialization*
- *New users*

One of the techniques that this type of RS uses to form the item profile is **Term frequency – Inverse Document Frequency (TF-IDF)** that enumerates the frequency of appearance of every word in a document and gives less weight to the words that are continuously repeating. After this task is completed the vector space model can be computed to obtain similarities between items with the angle between vectors (each vector is an item).

Knowledge-based (KB) appeared because of the lack of the two previous RS to adapt to every situation. KB is a technique that fits to the recommender case. There are two types inside KB: constraint-based and case-based. In both the users must specify its requirements so that the RS comes with a solution. Case-based solves new problems using data of old cases that have had an optimal result. Constraint-based is based in filters or preferences settled by the user. The development of constraint-based is not easy because the system has to adapt to every new filter the user wants to add.

Hybrid RS combine two or more types of RS to form a "total" system that mixes techniques. There are seven different hybridization methods described in Table 2.2.

RECOMMENDER SYSTEM WITH COLLABORATIVE FILTERING ALGORITHMS

Types of feedback for collaborative filtering

Ratings play an important role in every CF algorithm. This is the reason why this chapter starts with the explanation of the two types of ratings: explicit and implicit.

When the user rates with stars or points an item is giving **explicit data** to the recommender system because the rating that is going to figure in the RS is the one that the user gave to the item directly. On the other hand, **implicit feedback** is extracted by the behaviour of the user i.e. how many times did the user buy an item or what is the frequency with which the user buys a specific product.

Matrix factorization (MF)

MF consists in the factorization of a matrix with considerable dimensions into small matrices. The result of the product between these matrices will be the original big matrix. The goal of this reduction of dimension is to obtain **latent factors**. Latent factors

characterize each user and each item. FM techniques make a big effort to compute these latent factors and one of the most important algorithms to develop this task is **SVD**. It consists in decomposing an initial matrix into three small matrices. U is going to be the user's matrix, V is going to be the items matrix and Σ is a diagonal matrix for the singular values. After running SVD algorithm that separates the initial data into three. The main problem of SVD inside RS is that ratings matrix is very sparse, and this technique is not developed for a matrix in which a lot of positions are zeros. For this reason, SVD is not valid for RS.

Learning Algorithms

ALS is an iterative optimization algorithm, but it differs from SVD. The difference between them is that ALS does not recompute the product of both vectors p and q, it recomputes that vectors. This algorithm is used when the matrix is sparse because it does have a solution for the positions in which there is a zero. It fixes the items matrix and computes users matrix, when this process ends, it fixes users matrix and calculates items matrix. When ALS is integrated in RS it is always set a confidence level so that low ratings have more importance in the full system. The entire expression for ALS is equation 4.11 in which there is a regularization term that can be tuned by cross validation for better results. This value is always small in comparison with the ratings values because that way $\boldsymbol{\lambda}$ is going to penalize high values to avoid problems as overfitting.

Precision prediction measures

There are two type of metrics, one for each kind of feedback. Metrics for explicit feedback are **MAE**, **MSE** or **RMSE**. These three algorithms calculate the error between the predicted rating and the real one and divide it by the total number of elements. **MAE** (equation 4.17) does the absolute value of the error, **MSE** (equation 4.18) squares this error and **RMSE** (equation 4.19) does the square root of **MSE**. The three of them eliminate signs with the square or with an absolute value but MSE and **RMSE** are the ones that penalize high values with the squared error. The most useful metric is **RMSE** because it has the same efficiency of **MSE**, but its results can be more understandable because the error that was squared in **MSE** is now in the ratings scale because of the square root.

Metrics for implicit feedback are: **mAP** and **nDCG**. These measures need **precision** and **recall**. Precision is the percentage of elements that are relevant inside the set of selected elements. The goal of precision is that the user only selects the relevant items and not any more. It assumes that there are more relevant items than the RS needs. Recall is the percentage of relevant elements that are selected inside the relevant set. The goal is not to lose items that can be useful for the RS. There is a measure that combines these two: **F1-score**.

Now that precision and recall have been exposed, mAP and nDCG can be explained.
mAP will measure the efficiency of the entire system and nDCG will measure the similarity of the recommended list obtained with the ideal case in which every recommended item is relevant to the user.

AP is the sum of the precision of relevant elements and divided by the number of relevant items. By summing these AP's and dividing them by the number of users, mAP (equation 4.20) is obtained.

nDCG equation (4.22) is the normalized version of DCG (DCG divided by the ideal DCG). In DCG (equation 4.21) there are scores for each item according to its position in the rank. If one item has a low position, then its DCG has to be lower. This can be done because the equation of DCG is modelled by a factor $\frac{1}{\log(rank+1)}$. The principal difference between mAP and nDCG is that the second one has a smoother slope than mAP, as shown in Figure 4.19.

DATABASE ANALYSIS

Inside CF analysis there is an extent research about the user behaviour, focusing on how users do the order, what day of the week and amount of order done by each user. It has special importance the frequency of reordering products because afterwards that is the way the RS is going to obtain the ratings of every user for each product. To know more about the database there are several graphics about the way the items are distributed in each aisle or in each department.

When taking about CB analysis it must be noticed that the most important analysis has to be done in items. Every CB needs a good item feature profile. In this project every item has aisle, department, product id, product name and user that bought it. This is the reason why Figure 3.10 has been done, to know more about the distribution of items in the whole supermarket. The database has been separated into three different sets to show how well does the RS suggest products to users that the system cannot see when training the algorithm.

DEVELOPING A RECOMMENDER SYSTEM WITH COLLABORATIVE FILTERING

With this figure it can be explained the entire functionality of the RS. Firstly, the train set is used to train ALS algorithm. After this training the first results over validation set can be computed. These results are used to tune hyperparameters such as regularization, latent factors or iterations inside ALS.

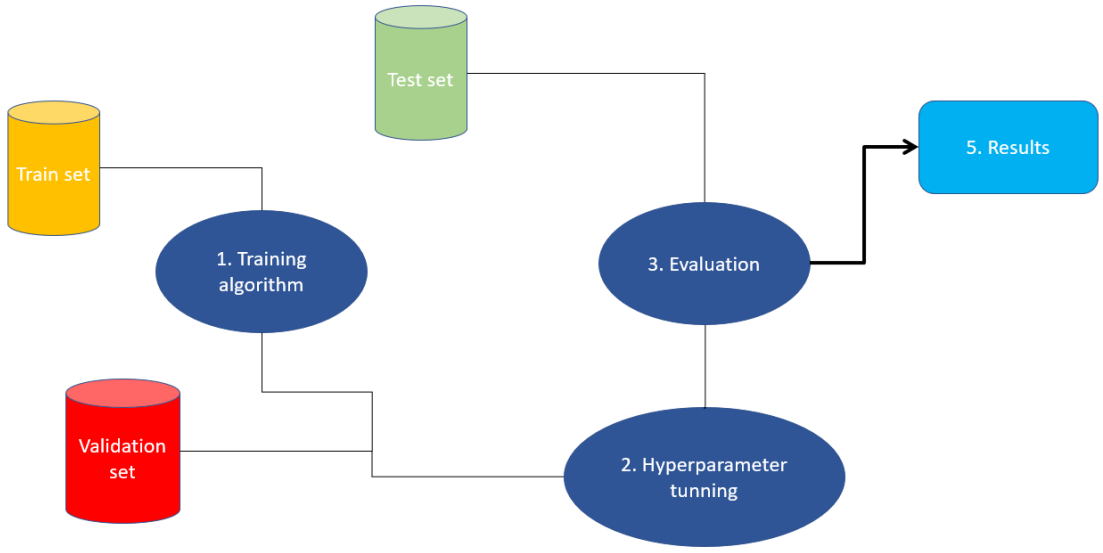

Fig. 8.1. *Functionality diagram of the RS algorithm*

The tests done over validation set indicate that the best parameters are: 130 latent factors, 0.05 as regularization parameter and 25 iterations over ALS. After computing these tests, some experiments have been done. These experiments over validation show that 40 % of the recommended items are relevant to the user.

When these results are obtained over test set, measuring the real performance of the system, the efficiency of the system decreases until 10 %. This drawback is caused by overfitting because the validation and test results are highly different. There are two users that are more active than the others because they purchase more frequently every item on its orders. The performance for these two users is significantly higher than the general case, 75 % of similarity with an ideal recommender list.

To conclude this chapter, ALS implemented with Implicit Python library has also the functionality to recommend similar items based on its name, similar aisle or similar department. There is an example of three recommendations for one product in Figure 5.10.

PROJECT MANAGEMENT

The project was done in two main tasks: theory and practice. Inside the first one, a deep study of the RS and its algorithms was made. According to the second task, this was the one that was more tough because of the complexity of working with a new programming language as Python.

The budget of this project is divided into: staff, material including computers and the rent of a working space for the team. The total cost of the project is 15.000€ without considering the maintenance of the system because that is a matter of the company that purchases the system.

CONCLUSIONS

To conclude, the main objectives of this project had been fulfilled. The RS has better results than a random model. There is an overfitting problem because the difference between validation and test results is considerable. To improve this, one of the future lines proposed is cross validation to tune hyperparameters.

The end of this project that was to develop a RS based on CF with an algorithm manually made, is reached because the results are better than a random model.